Master's thesis

NTNU
Norwegian University of Science and Technology
Faculty of Natural Sciences
Department of Materials Science and Engineering

Per Fredrik Daun

A Comparative Study of Different CFD-codes for Fluidized Beds

Master's thesis in Material Science and Engineering
Supervisor: Kristian Etienne Einarsrud
June 2023

NTNU
Norwegian University of
Science and Technology

Per Fredrik Daun

A Comparative Study of Different CFD-codes for Fluidized Beds

Master's thesis in Material Science and Engineering
Supervisor: Kristian Etienne Einarsrud
June 2023

Norwegian University of Science and Technology
Faculty of Natural Sciences
Department of Materials Science and Engineering

NTNU
Norwegian University of
Science and Technology

Preface

This thesis is submitted as a requirement for the course TMT4905 - Materials Technology, Master's Thesis, to the Norwegian University of Science and Technology (NTNU)

The present work serves as the master thesis of the author and was carried out at the Department of Material Science and Engineering at NTNU, Gløshaugen during the spring of 2023. This project was supervised by Professor Kristian Etienne Einarsrud.

"The roots of education are bitter,
but the fruit is sweet"
— Aristotle

Acknowledgments

First of all, I would like to thank my supervisor Kristian who has been a great help throughout this entire project. The weekly meetings, in addition to my running down to his office for quick guidance, have proven to be an invaluable resource throughout the project. His pleasant demeanour and rapid feedback has made working on the project an enjoyable experience.

I would like to thank my mother and father for always believing in me and giving me words of encouragement throughout the entire duration of my studies at NTNU.

Lastly, I would like to thank my fellow students, who have made the last 5 years of my life the most enjoyable so far. I would especially like to thank those I have been seated together with for the last year as they have helped keep my spirits/morale high, in addition to providing good feedback on how to structure and present my thesis.

Abstract

Fluidized beds are important processes with many applications in different industries. One of its uses are as a dry scrubber for processing the off-gas from aluminium production. Increased demand for aluminium in the world means that more off-gas is produced and thus more sophisticated and effective dry scrubbers are needed. It is necessary to know how the hydrodynamics of a fluidized bed function and look if one is to be able to properly design a fluidized bed reactor. One way to attain knowledge of the hydrodynamics is by utilizing computational fluid dynamics (CFD) which can give a general overview of the processes without the need for large and expensive experimental setups. It is however necessary to validate the CFD models and software if one is to be able to trust the results it provides.

This thesis therefore attempts to validate the OpenFOAM solver *multiphaseEulerFoam* for an Eulerian-Eulerian two-fluid model with the Syamlal-O'Brien and Gidaspow drag models at varying superficial gas velocities by comparing the results it provides with the already validated software MFiX and with experimental data. The present findings find OpenFOAM to be a mature predictor in relation to the fluidization phenomenon and it produces results that are in reasonable agreement with both MFiX and experimental data. Further study of the *multiphaseEulerFoam* solver is however needed to understand further how different settings affect it. The findings also show that the Gidaspow and Syamlal-O'Brien drag models are good for predicting fluidization, but under different conditions, where the former excels at low velocities and the latter at high velocities.

Sammendrag

Fluidiserte senger er viktige prosesser med mange bruksområder i ulike industrier. Ett av bruksområdene er som en tørrskrubber for å behandle avgass fra aluminiumsproduksjon. Økt etterspørsel etter aluminium i verden betyr at det produseres mer gass, og dermed trengs det mer sofistikerte og effektive tørrskrubbere. Det er nødvendig å forstå hydrodynamikken i en fluidisert seng og hvordan den fungerer for å kunne designe en god og effektiv fluidisert seng reaktor på riktig måte. En måte å tilegne kunnskap om hydrodynamikken er ved å bruke numerisk fluid dynamikk (CFD), som kan gi en generell oversikt over fluidiseringsfenomenet uten behov for store og dyre eksperimentelle oppsett. Det er imidlertid nødvendig å validere CFD-modeller og programvare for å kunne stole på resultatene de gir.

Denne avhandlingen prøver derfor å validere OpenFOAM-løseren *multiphaseEulerFoam* for en Eulerisk-Eulerisk to fluids modell med Syamlal-O'Brien og Gidaspow-dragmodellene ved varierende inngangs gasshastigheter, ved å sammenligne resultatene den gir med den allerede validerte programvaren MFiX og eksperimentelle data. De foreliggende funnene viser at OpenFOAM er en moden prediktor i forhold til fluidiseringfenomenet, og den produserer resultater som er i rimelig overensstemmelse med både MFiX og eksperimentelle data. Videre studie av *multiphaseEulerFoam*-løseren er imidlertid nødvendig for å bedre forstå hvordan ulike innstillinger påvirker den. Funnene viser også at Gidaspow- og Syamlal-O'Brien-dragmodellene er gode for å forutsi fluidiseringfenomenet, men under ulike betingelser, der den første er best ved lave hastigheter og den andre ved høye hastigheter.

Table of Contents

List of Figures

List of Tables

Listings

1 Introduction

1.1 Aluminium production

Aluminium is the second most produced metal in the world, beaten only by the steel industry, and the demand for aluminium is projected to increase by over 35% by 2030. This is due to the fact that aluminium is a versatile material with many different applications like for example in the transport and electrical sectors [1]. It is therefore necessary to improve the production processes to be more efficient so that this growth can happen sustainably.

Aluminium is produced in an electrolytic process called the Hall-Héroult process, named after its inventors, where alumina (Al_2O_3) is reduced to produce aluminium (Al). This is done by dissolving the alumina in a molten salt bath consisting of mainly cryolite (Na_3AlF_6) but also calcium fluoride (CaF_2) and aluminium fluoride (AlF_3). The salt melt contains large amounts of fluoride and this is so that sufficient alumina dissolution can occur [2]. Electricity is supplied to the cell by a direct current that flows from a carbon anode to a cathode at the bottom of the cell. This current supplies the energy required for the reduction reaction in addition to generating heat due to the resistivity of the melt. The anodes are consumed in the reduction process, which results in the formation of carbon dioxide (CO_2) and some carbon monoxide (CO) [3]. The net reaction for the aluminium production is given as:

$$\frac{1}{2}Al_2O_3(s) + \frac{3}{4}C(s) = Al(l) + \frac{3}{4}CO_2(g). \tag{1.1}$$

The produced metal has a higher density than the molten bath and sinks to the bottom of the cell. This layer of produced aluminium is tapped regularly at an interval of once per day. The amount tapped is roughly equal to the amount that is averagely produced in a day, and a constant layer of aluminium is kept at the bottom of the cell. This layer of aluminium is kept in the cell to ensure stability as the cell reaction requires a constant anode-cathode distance to operate desirably. The entire bed is covered with an anode cover material (ACM) which forms a crust with solidified alumina and fluorides at the top of the cell. This is done to prevent the anodes from reacting with the surrounding air, which consumes the anode from the top. The crust over the cell also works as a shield that prevents heat loss from the cell [4].

Modern electrolytic cells in aluminium production are contained within a structure, this allows for the containment and capture of the gases and debris that are generated so that it can be sent to a gas treatment centre and prevent the emissions of certain hazardous chemicals. A rough sketch of an aluminium production facility and the material flow can be seen in figure (1.1).

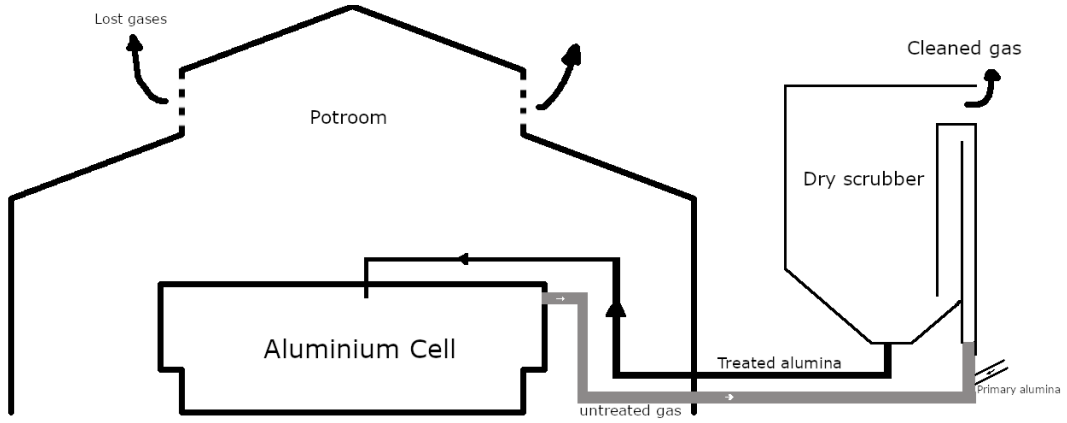


Figure 1.1: Schematic of aluminium production, with a gas treatment facility. The figure is inspired by [3]

1.2 Gas emissions and treatment

The composition of the off-gas that is created during aluminium production contains many different species, and the exact molecules present are difficult to determine. However it is possible to determine the main groups of species, which are; carbon oxides, sulfurous gas species and fluorides [3]. The main way of cleaning the produced off-gas is by utilizing a dry scrubber, which is a system that utilizes a dry compound to react and remove the undesired compounds from the gas. In the case of aluminium production, alumina is utilized to react with hydrogen fluoride (HF) to create aluminium fluoride, which can be recycled back into the cell. Modern dry scrubbers manage to scrub most of the HF from the pot gas (up to 99%), so the main issue of development is concerned with increasing the efficiency of the process.

One of the problems with developing new and more efficient dry scrubbing systems is that they are complex systems consisting of three main factors: kinetics, transport phenomena and flow pattern. This means that a complex system that appears to work on the lab scale likely won't scale up, as the different geometric scale alters one or more of the factors mentioned, and the transport phenomena at the small scale is entirely different on the industrial scale. Witt and Hickman recommend that reactors should be kept simple with fluids simulated at large scale so that the chances of a successful scale-up are as high as possible [5]. Figure (1.2) shows the necessary steps in order to go from lab-scale experiment to industrial scale as proposed by Witt and Hickman. This thesis is concerned with the transport behaviour, marked in red in the figure. This is due to the fact that the transport behaviour and hydrodynamics of the fluidized bed is of great importance when determining the quality of a potential reactor, and allows for a clearer choice of bed reactor type.

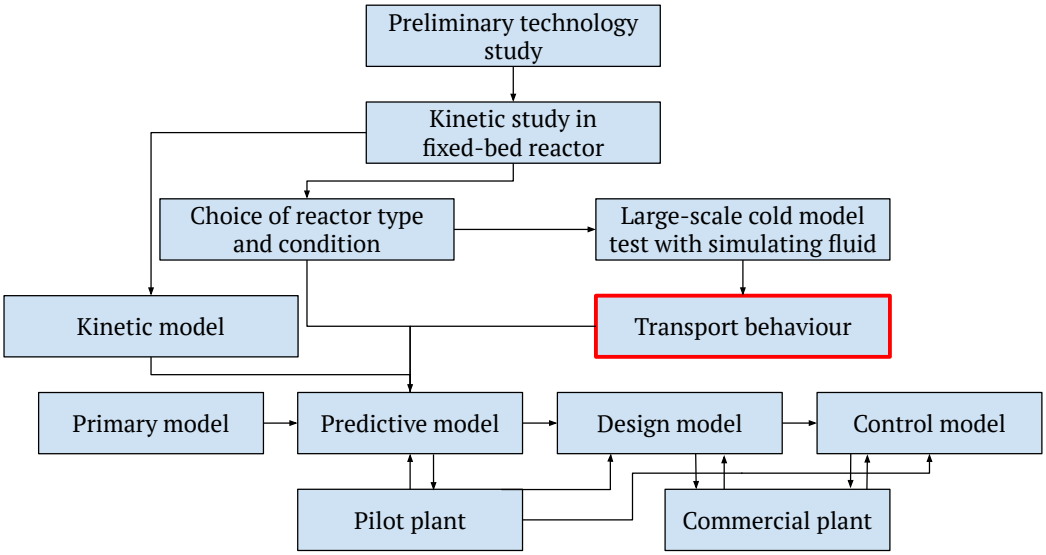


Figure 1.2: The scale up process for designing and developing new fluidized bed reactors. The step with the red border "Transport behaviour", is the one this thesis is concerned with. Figure is inspired by [5]

Because of this necessity for a transport model, it is paramount to know whether or not the simulation software is able to properly simulate the flow. One therefore needs to validate the simulation model. This is achieved by comparing gradually more complex simulated results with similar real world systems to see how right or wrong the model is [6]. Iteratively doing this allows one to find out which uncertainties are present in the model concerning a given physical phenomenon.

1.3 Goals and Aim

This thesis has several goals that can be seen presented in the following list

1. Review theory of packed bed and fluidized beds
2. Review existing literature on simulations that have been conducted on fluidized bed transport phenomena
3. Review the theory and implementation of computational fluid dynamics
4. Implement, validate and compare simulation capabilities of MFiX and OpenFOAM in the context of fluidized beds.

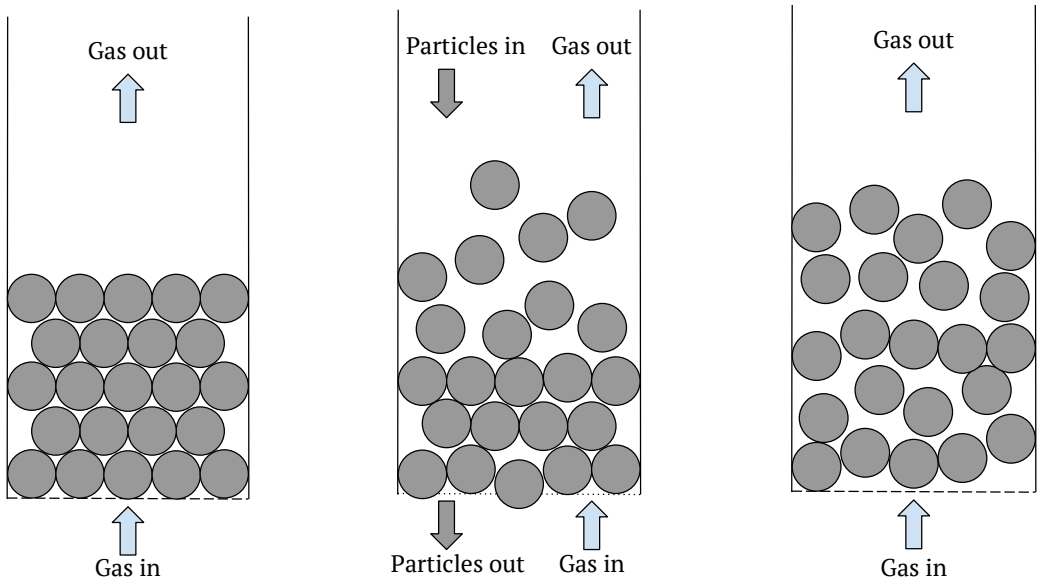
2 Review

The attraction of certain species in gases and liquids to solids is the basis for many types of separation technologies. Dry scrubbing, which is widely used in the aluminium industry is one example of this technology. The process involves a chemical component in a gas or a liquid called an adsorbate, adsorbing on the surface of a solid, called an adsorbant [7].

The aim of this chapter is to give an overview of the function of gas-solid systems, different types of reactor design, with their benefits and drawbacks. The general hydrodynamics of a gas-solid system will also be discussed. The coming discussions will be based on the book by Szekely et. al. [8].

2.1 Gas-solid reactors

The goal of a gas-solid reactor is to ensure sufficient contact and residence time between a large amount of solid particles and a continuously moving gas so that a complete reaction can happen between them. There are many ways to arrange multiparticle reactors some of which are fixed beds, moving beds, and fluidized beds among others. The choice of arrangement is of grave importance as it can significantly alter the reaction kinetics. A sketch of the mentioned reactor types can be seen in figure 2.1.



(a) Packed bed. (b) Moving bed. (c) Fluidized bed.

Figure 2.1: Sketch of different reactor configurations.

The main challenge with constructing a good gas-solid reactor is to relate the kinetics that can be obtained for a single particle to that of a multiparticle system. Although a particle's chemical properties remain unchanged in a multiparticle system with different flow regimes, the total kinetics of the system can be affected quite significantly. One must therefore be conscious of three main problems/differences when making these relations for a reactor.

The first problem revolves around the physical properties of the reactor assembly. The design of the reactor might depend critically on the expected pressure drop and one must thusly be intimately familiar with the relationship between the pressure drop and the fluid flow properties, furthermore, certain properties of the particles, such as the film coefficient could be significantly altered by interactions with other particles. The second problem that needs to be acknowledged is the residence time of the particles, meaning how long a particle spends in the reactor. For a

large system with many particles, it is only possible to know the average contact time between the gas and a given particle, this fact is especially true for fluidized or moving particle beds where rapid exchange of particles can occur. The third and final problem is the effect of the gas distribution, meaning that is impossible to control for a uniform gas composition and temperature in the entire reactor. This leads to there being reaction kinetics that differs from that of a single particle throughout the reactor. These three key differences and problems allow one to see that a multiparticle system is complex, and scaling up from the kinetic knowledge of a single particle is non trivial.

2.1.1 Fixed beds

Fixed bed reactors are characterised by the solid bed being arranged in a packed manner without noticeable movement among the solid particles as the gas passes through them. As a result of this packed arrangement a reaction front, that moves along the solid particles as the reaction with the gas occurs is formed. The reaction front phenomenon can be seen illustrated in figure (2.2). The reactor is typically either designed as a batch process, where an entire bed is reacted and then exchanged for a new one, or a bed that is continuously moving perpendicular to the gas stream, which is typically more efficient for a large scale industrial process.

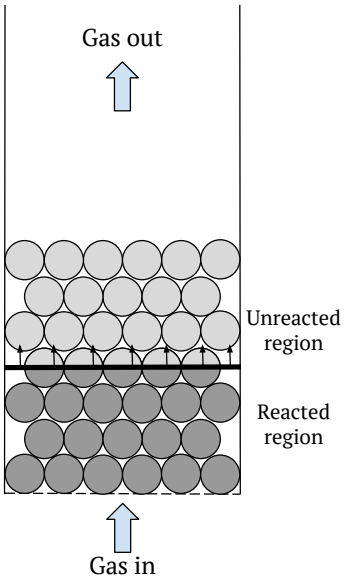


Figure 2.2: Reaction front moving along the particles with the gas stream. Dark gray indicates reacted material and light gray indicates unreacted material.

The fluid flow in a packed bed is typically described by the empirical relationship between the fluid flow rate and the pressure drop across the bed. This relation is typically described by the Ergun equation [9]:

$$\frac{\Delta P}{L} = 150\frac{(1-\alpha_g)^2}{\alpha_g}\frac{\mu_g \mathbf{u}_0}{(\phi d_s)^2} + 1.75\frac{(1-\alpha_g)}{\alpha_g^3}\frac{\rho_g \mathbf{u}_0^2}{\phi d_s} \tag{2.1}$$

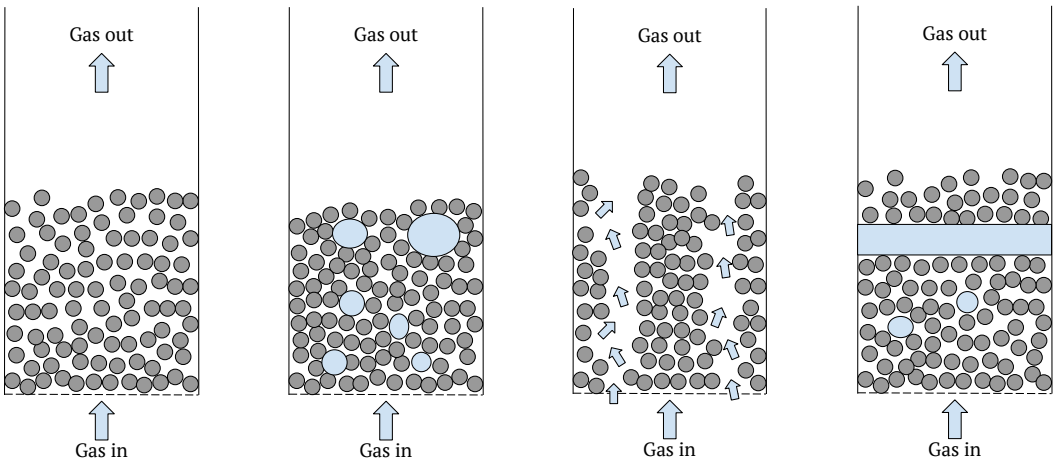
where ΔP is the pressure drop, L is the bed depth, α_g is the void fraction, μ_g is the fluid viscosity, ϕ is a shape factor, d_s the particle diameter, ρ_g the fluid density and $\mathbf{u}_0$ is the velocity of the incoming fluid. Some of the shortcomings of the Ergun equation are that it assumes some ideal conditions, such as an incompressible fluid, a spatially uniform packed bed, and that the system is isothermal. Due to the aforementioned shortcomings, one needs to be familiar with other non-ideal flow regimes that can occur in packed beds, such as preferential flows and dispersion.

A preferential flow is when the fluid travels through certain channels of the packed bed leading to an uneven distribution of contact between the particles and the gas. This phenomenon occurs due to uneven packing and beds containing particles of different sizes. It was also found that uneven packing leads to segregation in the bed and therefore higher porosity at the centre of the bed [10]. Pockets of higher or lower density throughout the bed do not only alter flow through themselves but the flow malalignment can also propagate to areas with regular packing. Axial and radial dispersion can occur due to eddy diffusion on a macro scale within the system, however, the effect of the dispersion is likely to be negligible in beds with reasonable bath depth and gas velocities.

2.1.2 Fluidization

Fluidized beds are often seen as an alluring alternative to fixed beds. This is due to the excellent mixing of solids that is achieved, which lowers temperature variation in the bed in addition to making the solids more easy to handle. The principle behind a fluidized bed is utilizing the velocity of the gas to suspend the particles in the stream, which is called fluidization. When a bed of particles is fluidized it acts like a well stirred boiling liquid and therefore allows it to readily be filled and tapped like a regular liquid, which allows for simple and rapid exchange of the solid material.

When the pressure drop across the bed is equal to the weight of the particles, they will begin to rearrange to the state of minimal packing. If the gas velocity is further increased beyond this point the pressure drop stays nearly constant and the particles will be suspended in the gas flow as the fluidized state is achieved. This is the ideal behaviour of a fluidized bed, there are however two commonly found deviations from ideality, slugging and channelling. Slugging occurs when there are large pockets of gas in the bed and periodic collapses of solid material into these voids cause fluctuations in the pressure drop across the bed. Channelling is similar to preferential flow in a packed bed, as only certain preferred channels become fluidized due to increased gas flow, while the rest of the bed stays packed. A sketch of 4 different states encountered in fluidized reactors can be seen in figure (2.3a). The first two sub-figures represent ideal and acceptable bubbling behaviour while the two latter ones represents the deviations from ideality explained previously.



(a) Fluidized bed. (b) Bubbling. (c) Channeling. (d) Slugging.

Figure 2.3: Sketches of various flow phenomena that can occur in a fluidized bed. Gray indicates particles and light blue indicates gas.

It is possible to determine the minimum required gas velocity to achieve fluidization by utilizing the relationship between the pressure drop and bed weight discussed previously, and relating it to the Ergun equation. The relationship between the weight of the bed and the pressure drop is described by:

$$\Delta P = L_{mf}(1-\alpha_{mf})g(\bar{\rho}_s - \bar{\rho}_g) \tag{2.2}$$

where L_{mf} is the height of the bed, α_{mf} is the minimum void fraction of the bed at the onset of fluidization, and $\bar{\rho}_s$ and $\bar{\rho}_g$ represent the average densities of the solids and gas respectively and g is gravity. This relation can now be inserted into Ergun equation, which yields:

$$\frac{1.75}{\phi\alpha_{mf}^3}(\frac{d_s u_{mf}\bar{\rho_g}}{\mu})^2 + \frac{150(1-\alpha_{mf})}{\phi^2\alpha_{mf}^3}(\frac{d_s u_{mf}\bar{\rho_g}}{\mu})^2 = \frac{d_s^3\bar{\rho}_g(\bar{\rho}_s - \bar{\rho}_g)g}{\mu^2} \tag{2.3}$$

which is a quadratic equation that can be utilized to determine the minimum fluidization velocity, u_{mf}. There exist several empirical models for determining the minimum fluidization velocity in case the minimum fluidization voidage is unknown. These models will be discussed in greater detail in section 3.

The opposite of the minimum fluidization velocity is elutriation velocity, which represents the upper limit on the gas velocity in a fluidized bed. The elutriation velocity represents the velocity where entrainment occurs i.e. the solid particles leave the reactor together with the gas. A typical approximation for the elutriation velocity is the terminal velocity of the particle, which is determined by equating the drag force of the fluid to the weight of the particle. The gas velocity is typically kept quite a bit lower than the elutriation velocity due to the variation in particle sizes typically encountered in an industrial process.

It has been found that an excess of gas, approximately equal to the amount required to maintain fluidization, passes through the bed as bubbles. There are many different ways of estimating the velocity of a single bubble in a fluidized bed, one of which was described by Davidson and Harrison [11]:

$$u_{br} = 0.79 g^{\frac{1}{2}} V_b^{\frac{1}{6}} \tag{2.4}$$

where u_{br} is the rising bubble velocity of a single bubble and V_b is the bubble volume. The maximum velocity a bubble can have in the stream is estimated to be roughly equal to the terminal velocity of the particles in the bed. This is due to particles being drawn into the wake of the bubble, which would cause it to burst. Bubbles travelling through the bed will have an effect on each other and David and Harrison found another equation to determine the bubble velocity under these conditions:

$$u_b = u_{br} + (u_0 - u_{mf}) \tag{2.5}$$

where u_b is the bubble velocity, u_{br} is the rising velocity of a single bubble in similar conditions and u_0 is the superficial gas velocity. This model for rising bubble velocity was tested against other more modern and complex models by Karimipour [12] and was found to be sufficiently accurate. Gas bubbles play an important role in fluidized beds, as they ensure excellent mixing, however there are drawbacks as the gas contained within them does not come into intimate contact with the solids. One must therefore account for gas exchange between the bubble- and mixed- 'phases'. This becomes an issue when 'intermediate' sized bubbles are present as these bubbles have layers of solid particles around them, which resists gas transfer between the bubble and the mixed phase.

The exact flow patterns of the gas in fluidized beds are not entirely known, and are difficult to predict. This is due to crossflow that is caused by the solid particle inlet and outlets. Experiments that have been done on fluidized beds have determined that the overall pattern lies somewhere between plug flow and complete mixing. This can be understood by the emulsified phase consisting of gas and solids that are completely mixed, while the bubbles move through the system in plug flow.

The effect of size distribution of the particles in the fluidized bed must not be neglected. Huilin et al. [13], studied the effects of minimum fluidization velocity and size segregation of particles in a

fluidized bed. Their findings indicated that particle diameter and the mass fractions of particles of a given diameter changed the pressure drop in the reactor which impacted the minimum fluidization velocity. Where lower pressure drop occurred for fluidized beds with a large fraction of small particles and vice versa. They also found that the size fractions tended to segregate with small particles rising to the top while larger particles remained at the bottom of the reactor. The segregation was found to increase as the difference in particle diameters increased.

2.2 Previous Simulations

The hydrodynamics of a fluidized bed are complex, and very susceptible to change due to slight variations of operating parameters. For this reason, the development and validation of computational fluid dynamics (CFD) simulations is of great interest, and many studies have been conducted trying to refine and assess the accuracy of the simulations and how certain parameters affect the model. Many simulations are conducted in 2D as opposed to 3D to save computational resources, this is done by only considering one plane in the computational domain, this however means that some of the properties of the flow are lost due to the neglection of depth. An example of this can be seen in figure (2.4), where the shaded red area represents the domain of a 2D simulation, while the entire box would be the same domain in a 3D simulation.

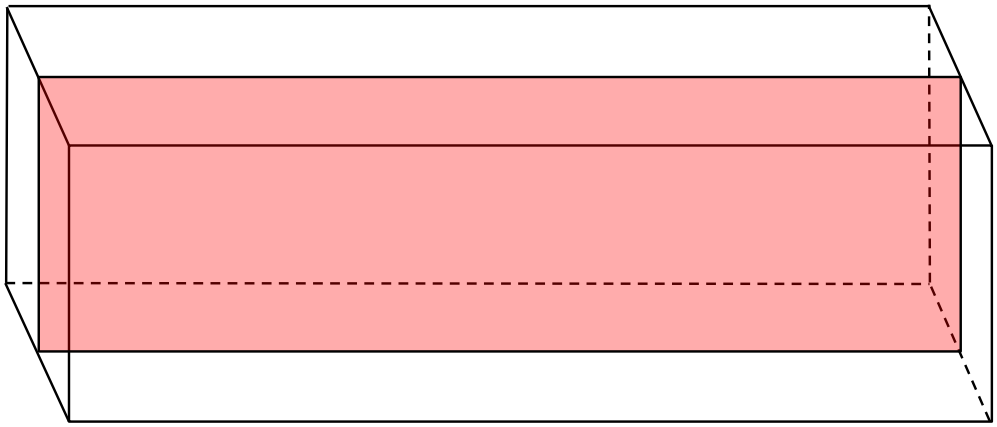


Figure 2.4: The entire box represents the 3D domain, while the shaded red area represents the plane that is computed when opting for a 2D simulation

The simulations conducted in this thesis will be based on the Eulerian-Eulerian model, also known as the two-fluid model. In this model both the gas and solid phases are considered as interpenetrating continua, this means that it does not need to monitor the motion and behaviour of every single particle, which significantly reduces the computational time. The model solves the governing equations for each phase separately together with interphase exchange terms that govern the exchange between the phases. The model typically incorporates the kinetic theory of granular flow (KTGF) to handle and calculate the necessary properties of the solid continuum [14].

Taghipour et al. [15] conducted a study where they compared experimental results for the hydrodynamics in a simple column fluidized bed with an identical computational model with the FLUENT 6.0 software. The experimental setup consisted of a pseudo-2D column of 1.0m height, 0.28m width and 0.025m depth. Their simulations were done in a 2D Eulerian-Eulerian (E-E) model, with the Gidaspow, Syamlal and Wen-Yu drag models. Their findings indicated that the bed reached steady state after 3 seconds, when the height expansion and bubble generation stabilized. They found that there was reasonable agreement between the drag models and the experimental result, with increasing bed expansion, pressure drop and voidage with increasing gas velocities. They did however find that the simulated result overpredicts the voidage and bed expansion and concluded that further efforts are needed to validate the CFD models.

Cornelissen et al. [16] studied the effects of mesh size, time step and convergence criteria in an E-E model of a liquid-solid fluidized bed utilizing FLUENT 6.1.22 by comparing with an experiment. They found that a convergence criterion of 10^{-4} yielded a convergent result significantly faster than 10^{-3}, which they described as the standard convergence criterion at the time. In their investigation of time step and mesh size, they found that both too high and too low Courant numbers yielded unrealistic results and found that $0.03 \leq N_c \leq 0.3$ yielded good results independent of mesh size and time step.

Li et al. [17] investigated the influence of the solid phase boundary conditions by simulating different values for the specularity and particle-wall restitution coefficients. The study was conducted by simulating the same geometry and velocity in both 2D and 3D E-E models in the MFIX software and finally comparing with a pseudo 2D experiment. Their findings indicated that the effect of the restitution coefficient was very small and negligible in comparison to the specularity coefficient. They found that specularity coefficients below 0.05 deviated significantly from all the other values tested with solids only flowing downwards close to the wall, therefore producing different circulation patterns. When comparing the 2D and 3D results with the experiment they found that both capture the general flow behaviour, however 3D simulations showed much better agreement. This indicated that the wall effect in a pseudo 2D experiment is of great importance when one is trying to validate a CFD model.

Herzog et al. [18] did a comparative study, where results obtained from MFIX, OpenFOAM 1.6/2.0 and FLUENT 6.3 for a 2D E-E model were compared with the experimental results obtained by Taghipour [15]. They tested the CFD codes' abilities to predict pressure drop, bed expansion, void fraction and solid velocity, with both the Gidaspow and Syamlal O'Brien drag models. Their findings indicated that both FLUENT and MFIX were in good agreement with the experiment for both drag models, while OpenFOAM yielded unsatisfactory results. They therefore concluded that OpenFOAM needed further development and validation before it could be considered mature enough to predict fluidization.

Loha et al. [19, 20] conducted two separate studies on the Johnson-Jackson wall boundary condition in a 2D fluidized bed simulation, the first one investigating the influence of the specularity coefficient and the second investigating the wall-particle restitution coefficient. The simulations were conducted as a 2D E-E model utilizing the Gidaspow drag model in the FLUENT software. Their investigation on the specularity coefficient found deviations in the results for specularity coefficients equal 0 and 0.01, this is due to free slip and near free slip, which increases the downward particle flow along the walls. This effect then lead to an increased downward motion in the centre of the bed due to circulating flows. The specularity coefficients tested above 0.1 all showed reasonable agreement with experimental results. Investigations of restitution coefficients between 1 and 0.85 showed increased bubble size and formation for decreasing values, with increasing particle velocities and void fractions. The largest deviation occurred for restitution coefficient equal to 1, representing perfectly elastic collisions, with no bubble formation and little to no axial velocity and constant void fraction across the bed. Their results indicated that coefficients 0.95 and 0.99 gave the best agreement with experimental results.

Kong et al. [21] investigated the effect of grid size, time step and wall boundary conditions in circulating fluidized beds for an E-E model with the FLUENT 13.0 software. Their investigations into time step and grid size showed that a time step of 10^{-4} and a grid size equal 15 times the particle diameter yielded the best results. Smaller grid sizes than this showed large fluctuations in the solid volume fraction, which was caused by an instability in the wavelength of perturbations of the solid phase, which coincided with previous studies. Their testing on the wall boundary conditions showed that the specularity coefficient had a much larger impact relative to the restitution coefficient. Their findings showed that increased values of both the specularity- and restitution-coefficient lead to lower accumulation of solids and increased lateral velocity at the wall.

Altantzis et al. [22] used a 3D E-E model in the MFIX software in conjunction with existing experimental data to investigate the effect of the front and back walls, and the specularity coefficient, when doing validation with data from a pseudo 2D experimental setups. Their findings indicated that the inclusion of the front and back walls were of paramount importance when validating the hydrodynamic model. They also found that the correct value of the specularity coefficient varied

depending on the superficial gas velocity, with increasing values of the coefficient for lower gas velocities, indicating that the specularity coefficient likely depends on the solid slip velocity. They also showed that the value of the specularity coefficient significantly altered the flow regime with large slugs and low circulation times for low values, and increasing bubble frequency and longer circulation time as the coefficient was increased.

Bakshi et al. [23] compared 3D E-E models of cylindrical fluidized beds in the MFIX software with different experimental results found in the literature. The aim of their study was to find the impact of the wall boundary condition as well as the choice of drag model. Their rigorous testing of specularity coefficients indicated that the correct value of the specularity coefficient lies within the range of [0.01, 0.3]. Variations of the specularity coefficient within this range lead to insignificant changes in the overall hydrodynamics suggesting that any value within the range is suitable for adequate prediction of most bubbling fluidization cases. It was also found that fluidization occurred faster with fewer bubbles through the centre as the bed diameter increased, showing that the bed hydrodynamics is affected by the wall area to bed volume ratio. Their comparison of the Gidaspow and Syamlal-O'Brien indicated that the former is more suited for lower superficial gas velocities $U/U_{mf} < 4$ while the latter is more suited for higher velocities. This is due to the Gidaspow model underpredicting the gas-solid drag force under fast bubbling and slugging conditions, leading to an overprediction of bubble diameter.

Shi et al. [24] conducted a study attempting to validate 2D and 3D models with the experimental data obtained by Taghipour et al. [15]. They used E-E models in the FLUENT 16.2 software and tested the influence of dimension, turbulent vs laminar flow regime, wall boundary conditions, mesh resolution and drag model. Their results showed that a grid size corresponding to roughly 18 times the particle diameter gave the best agreement with the experimental result and further refinement of the mesh resulted in a less accurate model. Negligible differences were detected for specularity and restitution coefficients between 0.1-0.9 and 0.85-0.99 respectively, indicating that values within these ranges are acceptable for this experimental setup. Turbulence did not play a significant role within the bed however it did affect the gas flow above the bed, their conclusion was therefore that a laminar model can be utilized for unreactive and isothermal beds, however, the RANS k-ϵ model should be used otherwise. The testing of the drag models showed that the Syamlal-O'Brien model gave better agreement with the experimental data, with the best model being Syamlal-O'Brien with the Johnson-Jackson boundary condition and utilizing the turbulent RANS model. When testing the difference between 2D and 3D they found significantly better agreement with the experimental data for the 3D model, it was however possible to find a combination of parameters that could yield good agreement for the 2D model, this could unfortunately be a misleading finding as this combination parameters is likely unrealistic. Their suggestion was therefore that validation of numerical models should be done in 3D and that 2D should only be used for sensitivity analyses when the model is already validated.

3 Theoretical approach

3.1 Multiphase flow

Most flows that occur in nature can be described as multiphase flows, as a flow seldom contains just a single phase. Multiphase flows typically refers to a fluid flow where more than one of the thermodynamic phases; gas, liquid and solid, are present either in a single- or multicomponent system [25]. Some simple examples of multiphase flows are a liquid flowing together with its vapour or solid particles dispersed in a gas flow.

The simplest way to describe a multiphase flow is to assume that the phases form a perfect mixture. If this assumption is made one only needs to average the properties of the phases, with respect to their fraction, and solve a single set of equations, essentially assuming it to be a single phase fluid. This is fine when the interactions between the bulk fluid and its boundaries are of interest. However, this method neglects interactions within the flow and more sophisticated models must be utilized if these interactions are of interest.

The main model to predict the motion of a multiphase flow described in this work will be the *Two-fluid model*. In the two-fluid model, the dispersed phase is considered as a separate continuous phase that interacts with the main continuous phase. This means that the conservation equations must be developed and solved for both of the present phases. The greatest challenge with this model comes from the coupling of the two flows, especially due to the transient boundaries between them, additionally, it is possible to see how the particles within the dispersed phase interact with itself and how this influences the overall flow [26].

3.1.1 Definitions

When dealing with a multiphase flow it is reasonable to assume that the dispersed particles are not all uniformly spherical and that the shapes and rotations of each unique particle could have a meaningful impact on the physics of the problem. However, accounting for all these details increases the complexity a tremendous amount [27]. It is therefore more common to model each component based on its respective volume fraction given as:

$$\alpha_k = \frac{V_k}{V_{tot}} \tag{3.1}$$

where α_k is the volume fraction of the k-th phase, V_k is the volume occupied by the k-th phase and V_{tot} is the total volume. This is a quite intuitive method of defining the fraction of the different phases, however, it alone does not tell us about the particular properties or spatial arrangements of the different fractions, as can be seen in figure (3.1).

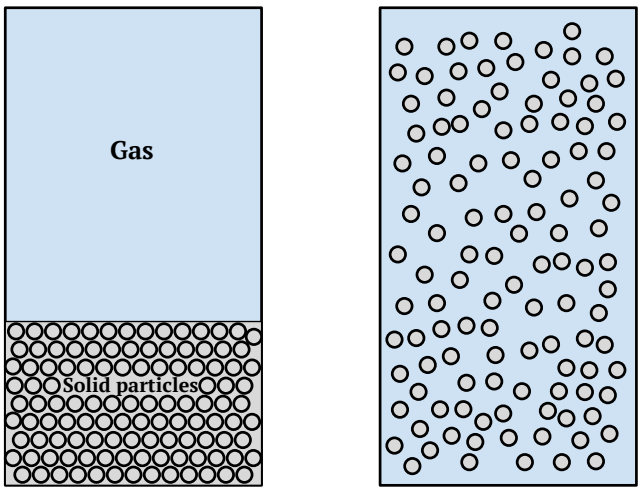


Figure 3.1: Equal volumes with the same number of dispersed particles but with different distribution and local void fraction. Closed packed particles under a gas (left), dispersed solid particles in a gas flow (right)

The sum of all phase fractions present in fluid flow must be equal to one which gives,

$$\sum_k \alpha_k = 1. \tag{3.2}$$

A multiphase flow containing gas with dispersed solid particles, as seen in figure (3.1), will contain the phases α_g and α_s representing the volumetric gas and solid fractions respectively. Inserting this into equation (3.2) yields

$$\alpha_g + \alpha_s = 1. \tag{3.3}$$

This then allows us to easily find the gas fraction, typically called the *void fraction*, for a given volume or area within the flow.

3.1.2 Balance Equations

The mass density for a given component or phase k, in a multiphase flow, can be defined in two different ways, both of which are useful in different circumstances [27]. One of the definitions stems from the mass of the component with respect to the volume it occupies, given as

$$m_k = \int \underline{\rho_k}\, dv \tag{3.4}$$

where $\underline{\rho_k}$ represents the mass of the k-th phase per total unit volume. This is similar to how it is also defined for a single component, however in a multiphase flow the component only occupies a fraction, α_k of the total volume, and the mass density is defined by

$$\alpha_k \rho_k = \underline{\rho_k}. \tag{3.5}$$

It is possible to create balance equations comparable to those of single phase flows for each of the k phases present in the flow, which will be presented next.

Mass balance

The main equation for mass balance for a component k is

$$\frac{d}{dt}\int \underline{\rho_k}\, dv = \int \Gamma_k\, dv \tag{3.6}$$

where Γ_k is a source term due to a different component transforming to component k, this could be either due to phase change or a chemical reaction. Rearranging the terms and removing the integrand we arrive at the continuity equation

$$\frac{\partial \rho_k}{\partial t} + \nabla \cdot (\underline{\rho_k}\mathbf{u_k}) = \Gamma_k \tag{3.7}$$

by inserting equation 3.5 into 3.7 we obtain

$$\frac{\partial(\alpha_k \rho_k)}{\partial t} + \nabla \cdot (\alpha_k \rho_k \mathbf{u_k}) = \Gamma_k. \tag{3.8}$$

Momentum balance

The balance of momentum for a component k is described by Newton's 2nd law

$$\frac{d}{dt}\int \underline{\rho_k}\mathbf{u_k}\, dv = \mathbf{F_k^b} + \mathbf{F_k^s} \tag{3.9}$$

where $\mathbf{F_k^b}$ and $\mathbf{F_k^b}$ are the source terms of the body and surface forces respectively. The source term for the body forces can be expressed as

$$\mathbf{F_k^b} = \int (\underline{\rho_k}\mathbf{b_k} + \mathbf{M_k i} + \Gamma_k \mathbf{u_{ki}})\, dv. \tag{3.10}$$

Where $\mathbf{b_k}$ represents the body force caused by external forces per unit mass, gravity is an example of such a force. The term $\mathbf{M_k}$ represents the forces acting on the k-th phase due to interactions with either the other phases present or with particles of the same phase. And $\Gamma_{ki}\mathbf{u_{ki}}$ represents a source of momentum due to mass sources. The surface forces $\mathbf{F_k^s}$ are represented as a integral over the stress tensor $\underline{\tau_k}$ at the surface of the phase

$$\mathbf{F_k^s} = \oint \underline{\tau_k}\, dS. \tag{3.11}$$

We can now insert equations 3.10 and 3.11 into 3.9, from which one obtains

$$\frac{d}{dt}\int \underline{\rho_k}\mathbf{u_k}\, dv = \oint \underline{\tau_k}\, dS + \int (\underline{\rho_k}\mathbf{b_k} + \mathbf{M_k} + \Gamma_k \mathbf{u_{ki}})\, dv. \tag{3.12}$$

By rearranging the terms, utilizing the divergence theorem and utilizing the continuity equation obtained previously (equation 3.8) gives the following equation

$$\int (\underline{\rho_k}\frac{d\mathbf{u_k}}{dt} + \Gamma_k(\mathbf{u_k} - \mathbf{u_{ki}}) - \nabla \cdot \underline{\tau_k} - \underline{\rho_k}\mathbf{b_k} - \mathbf{M_k})\, dv = 0. \tag{3.13}$$

Assuming that the equation is sufficiently continuous, the integrand can be removed and one obtains the differential form of the equation given as

$$\underline{\rho_k}\frac{d\mathbf{u_k}}{dt} = \Gamma_k(\mathbf{u_k i} - \mathbf{u_k}) - \nabla \cdot \underline{\tau_k} - \underline{\rho_k}\mathbf{b_k} - \mathbf{M_k}. \tag{3.14}$$

This equation can be rewritten by expanding the total derivative and invoking the continuity equation, in addition to utilizing the phase fractions in place of the variables that are dependent on the total volume. By doing these operations one finds the final form of the mass balance equation for a given phase k.

$$\frac{\partial \alpha_k \rho_k \mathbf{u_k}}{\partial t} + \nabla \cdot \alpha_k \rho_k \mathbf{u_k} \mathbf{u_k} = \Gamma_k \mathbf{u_{ki}} + \nabla \cdot \underline{\tau_k} + \alpha_k \rho_k \mathbf{b_k} + \mathbf{M_k}. \tag{3.15}$$

3.1.3 Forces

The body and surface forces, $\mathbf{F_k^b}$ and $\mathbf{F_k^s}$ were briefly described, but not expanded upon in the previous chapter and will be explored here. These forces are of great importance as they couple the phases together and can give insight into how they interact and show how these interactions influence the total flow. Additionally, there are some forces that only occur within a given phase, and will be different if the phase is gas or solid, these forces will allow one to study how intraphasial forces, especially within the solid phase, influence the dynamics of the flow. The equations and derivations in this subchapter are taken from the work of Zhong et al. [14].

The first force to be explored will be the surface force $\mathbf{F_k^s}$, which was shown to be represented by the stress tensor at the phase surfaces. Expanding the tensors for both the gas and solid phases yields [14].

$$\underline{\tau}_{g,ij} = \alpha_g \mu_g (\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_j}) - \frac{2}{3}\mu_g \delta_{ij} \frac{\partial u_k}{\partial x_k} \tag{3.16}$$

$$\underline{\tau}_{s,ij} = \alpha_s \mu_s (\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_j}) - \frac{2}{3}\mu_s \delta_{ij} \frac{\partial u_k}{\partial x_k} + \lambda_s \frac{\partial u_k}{\partial x_k} \tag{3.17}$$

where μ is the dynamic viscosity for the phases, and λ_s is the bulk viscosity, which is only present for the solid phase. The second term that is going to be expanded is $\mathbf{M_k}$, and will describe the pressure gradient within the phase as well as the drag force between the phases.

$$M_g = -\alpha \nabla P_g - \beta(\mathbf{u}_g - \mathbf{u}_s) \tag{3.18}$$

$$M_s = -\alpha \nabla P_s - \beta(\mathbf{u}_g - \mathbf{u}_s) \tag{3.19}$$

where P_g is the gas pressure, β is the interphase drag coefficient between the gas and the solid, and P_s represents the particle phase pressure, which is induced by collisions between the particles. Both the solid viscosities, μ_s and λ_s, that were introduced in the solid stress tensor, and the solid pressure are concepts within the kinetic theory of granular flow [28].

The kinetic theory of granular flow is used to create functions and equations that bring closure to the conservation equations in an Eulerian-Eulerian system. What is meant by closure equations is that they convert unknown variables into equations that contain known variables, this means that the amount of variables and equations become the same, and the set of equations becomes solvable. In the context of a gas-particle flow, this is done by considering the conservation of the solid fluctuation energy, which accounts for the kinetic and collisional influence of the particles [29]. The solid pressure P_s is given by the equation:

$$P_s = \alpha_s \rho_s \Theta_s + 2\rho_s(1+e)\alpha_s^2 g_0 \Theta_s \tag{3.20}$$

where e is the restitution coefficient, which represents how much velocity is kept when particles collide. Θ_s is the granular temperature, and g_0 is the radial distribution function which was defined by Carnahan and Starling [30] and is given by

$$g_0 = \left[1 - (\frac{\alpha_s}{\alpha_{s,max}})^{\frac{1}{3}}\right]^{-1} \tag{3.21}$$

which expresses the statistics of the spatial arrangement of the particles in the flow, meaning it can help to describe differences in density within the "continuous" particle phase.

The bulk viscosity, which was described by Lun et al. [31] is expressed as

$$\lambda_s = \frac{4}{5}\alpha_s\rho_s d_s g_0(1-e)(\frac{\Theta_s}{\pi})^{\frac{1}{2}} \tag{3.22}$$

and the solid shear viscosity μ_s is given as

$$\mu_s = \frac{4}{5}\alpha_s\rho_s d_s g_0(1+e)\sqrt{\frac{\Theta_s}{\pi}} + \frac{10\rho_s d_s\sqrt{\pi\Theta_s}}{96(1+e)\alpha_s g_0}\left[1+\frac{4}{5}\alpha_s g_0(1+e)\right]^2 + \frac{P_s\sin\Omega}{\sqrt{I_{2D}}} \tag{3.23}$$

where d_s is the particle diameter and Ω is the angle of internal friction. In both equations (3.22) and (3.23) we have introduced the granular temperature Θ_s for the solid particles, which is equivalent to the thermodynamic temperature for the gas. The granular temperature is estimated by the fluctuating kinetic energy of the particles

$$\Theta_s = \frac{1}{3}(u_s' u_s'). \tag{3.24}$$

The conservative equation for the particles' fluctuating energy is defined by

$$\frac{3}{2}\left[\frac{\partial}{\partial t}(\alpha_s\rho_s\Theta_s) + \nabla\cdot(\alpha\rho_s\mathbf{u}_s\Theta_s)\right] = -(P_s I + \alpha_s\tau_s) : \nabla\mathbf{u}_s + \nabla\cdot k_s\nabla\Theta_s - \gamma \tag{3.25}$$

where $(P_s I) + \alpha_s\tau_s : \nabla\mathbf{u}_s$ represents the fluctuating energy that is generated due to particle shear stress. $\nabla\cdot(k_s\nabla\Theta_s)$ and γ represent the conduction of fluctuating energy and the dissipation of fluctuating energy due to inelastic collisions respectively. Where the newly introduced unknown variables k_s and γ can be expressed as

$$k_s = \frac{2k_{dil}}{(1+e)g_0}\left[1+\frac{6}{5}(1+e)g_0\alpha_s\right]^2 + 2\alpha_s^2\rho_s d_s(\frac{\Theta_s}{\pi})^{\frac{1}{2}} \tag{3.26}$$

where k_{dil} is given by

$$k_{dil} = \frac{75\sqrt{\pi}}{384}\rho_s d_s\Theta_s^{\frac{1}{2}} \tag{3.27}$$

and γ is given as

$$\gamma = 3(1-e^2)\alpha_s\rho_s g_0\Theta_s\left[\frac{4}{d_s}(\frac{\Theta_s}{\pi})^{\frac{1}{2}} - \nabla\cdot\mathbf{u}_s\right]. \tag{3.28}$$

The particles in granular flow experience something called partial slip, this means that there is neither free slip (no friction at the boundary) nor no slip (zero velocity at the boundary), but

rather something in between. Johnson and Jackson [32] defined boundary conditions for the solid phase velocity and granular temperature which fulfil these criteria:

$$\mu_s \frac{\partial \mathbf{u}_s}{\partial x} = \frac{\pi \phi_s \rho_s \alpha_s g_0 \sqrt{\Theta_s}}{2\sqrt{3}\alpha_s^{max}} \mathbf{u}_s \tag{3.29}$$

$$\kappa_s \frac{\partial \Theta_s}{\partial x} = -\frac{\pi \phi_s \mathbf{u}_s^2 \rho_s \alpha_s g_0 \sqrt{\Theta_s}}{2\sqrt{3}\alpha_s^{max}} - \frac{\pi\sqrt{3}\rho_s \alpha_s g_0 (1-e^2)\sqrt{\Theta_s}}{4\alpha_s^{max}} \Theta_s \tag{3.30}$$

where μ_s is the solid shear viscosity, κ_s is the solid conductivity. ϕ_s and e are the specularity coefficient and restitution coefficients respectively. The specularity and restitution coefficients can only have values between 0 and 1. A value of 0 represents full slip condition for the specularity coefficient, and fully inelastic collision for the restitution coefficient, while a value of 1, means no-slip and a fully elastic collision.

Drag

Drag force can be looked at as the friction, or fluid resistance, between a solid body and the fluid it is moving through [33]. The inter-phase drag is the main phenomenon that couples the fluid and solid phases together in a multiphase problem, and the choice of drag model is what significantly influences the flow on a macroscopic scale [18]. The two models that will be utilized and explored in this thesis are the Gidaspow and Syamlal-O'Brien drag models.

The Gidaspow model, which was described by Ding and Gidaspow in 1991 [34], combines the Ergun equation with the Wen-Yu model. The Wen-Yu model is utilized if the particle flow is dilute, which is defined as a void fraction greater than 0.8. If the particle flow is more closely packed a modified version of the Ergun equation is used, which is obtained by comparing the Ergun equation to the gas momentum balance with no acceleration, wall friction or gravity and solving for the drag coefficient β [28]. The following model is then developed:

$$\beta = \begin{cases} \frac{3}{4}\frac{\alpha_s \alpha_g \rho_g |\mathbf{u}_g - \mathbf{u}_s|}{d_s} C_D \alpha_g^{-2.65}, & \alpha_g > 0.8. \\ 150\frac{\alpha_s^2 \mu_g}{\alpha_g d_s^2} + 1.75\frac{\rho_g \alpha_s |\mathbf{u}_g - \mathbf{u}_s|}{d_s}, & \alpha_g \leq 0.8. \end{cases} \tag{3.31}$$

where C_D is defined as

$$C_D = \begin{cases} 0.44, & Re_s \geq 1000. \\ \frac{24}{\alpha_s Re_s}(1 + 0.15(\alpha_g Re_s)^{0.687}, & Re_s < 1000. \end{cases} \tag{3.32}$$

and the Reynolds number is defined as

$$Re_s = \frac{\alpha_g \rho_g |\mathbf{u}_g - \mathbf{u}_s| d_s}{\mu_s}. \tag{3.33}$$

The Syamlal-O'Brien model was defined by Syamlal and O'Brien in 1988 and seeks to model the phenomenon of granular layer inversion in a fluidized bed. This layer inversion means that particles of different sizes and densities settle in different parts of the flow depending on the incoming fluid velocity. The model was described by showing that the drag coefficient of the particle phase can be related to the drag coefficient for an isolated sphere. This is done by utilizing the ratio between the terminal velocities of a group of particles and that of the singularly isolated particle. This ratio is described through empirical relations and experimental data [35]. The following model is found:

$$\beta = \frac{\frac{3}{4}\alpha_s \alpha_g \rho_g |\mathbf{u}_g - \mathbf{u}_s|}{u_{r,s}^2 \cdot d_s C_D(\frac{Re_s}{u_{r,s}})}, \tag{3.34}$$

where C_D is expressed by:

$$C_D = (0.63 + \frac{4.8}{\sqrt{Re_s/u_{r,s}}})^2 \tag{3.35}$$

with the same definition of the Reynolds number as in the aforementioned Gidaspow model. $\mathbf{u}_{r,s}$ is the ratio between the terminal velocities and is given as:

$$\mathbf{u}_{r,s} = 0.5\{A - 0.06Re_s + \sqrt{[0.0036Re_s^2 + 0.12Re_s(2B - A) + A^2]}\} \tag{3.36}$$

where A is defined as:

$$A = \alpha_g^{4.14} \tag{3.37}$$

and B is defined as:

$$B = \begin{cases} 0.8\alpha_g^{1.28}, & \alpha_g \leq 0.85. \\ \alpha_g^{2.65}, & \alpha_g > 0.85. \end{cases} \tag{3.38}$$

where 0.8 and 2.65 are calibration values that determine the minimum fluidization velocity of the granular particles.

4 Computational fluid dynamics

Fluid flow occurs when a fluid is exposed to external forces such as gravity, pressure and shear forces. Applying the fundamental laws of mechanics to a fluid continuum, in an attempt to describe the physical behaviour of the fluid, yields a set of partial differential equations (PDE) as was discussed in detail in the previous section. These equations are in most cases impossible to solve analytically, it is therefore necessary to utilize numerical methods, to create approximations of the equations so that they can be solved by a computer. This process is broadly speaking the subject matter within the field of computational fluid dynamics (CFD).

4.1 Discretization

Discretization is the process of splitting the exact and continuous solution to a PDE into a discrete domain that only contains variables at discrete locations in space and time. This allows for the conversion of the partial differential equation into a system of algebraic equations that can then be solved numerically. Discretization, and how it is implemented, is therefore of great importance in CFD as it allows for the computation of the governing equations and relevant flow variables throughout the fluid domain. There are many different methods to discretize a set of PDEs for example the finite volume method (FVM) and the finite element method (FEM) [36].

4.1.1 Finite Volume method

The finite volume method is one of the most popular methods of discretization within CFD due to its high flexibility. This flexibility stems from the fact that the discretization is done directly in the physical space, meaning that there is no need for transformation between the physical and computational coordinate system, in addition to its natural handling of the conservation properties in fluid dynamics [37].

The first step in the FVM is to discretize the solution domain into a computational domain. This is done by splitting the desired geometry into a grid system of non-overlapping sub-domains that are typically called control volumes (CV) or cells. Each control volume has defined boundaries, called faces, wherein conservation is applied and a node in the centre where the calculation takes place. A sketch of how a domain can be discretized, with CV nomenclature can be seen in figure (4.1)

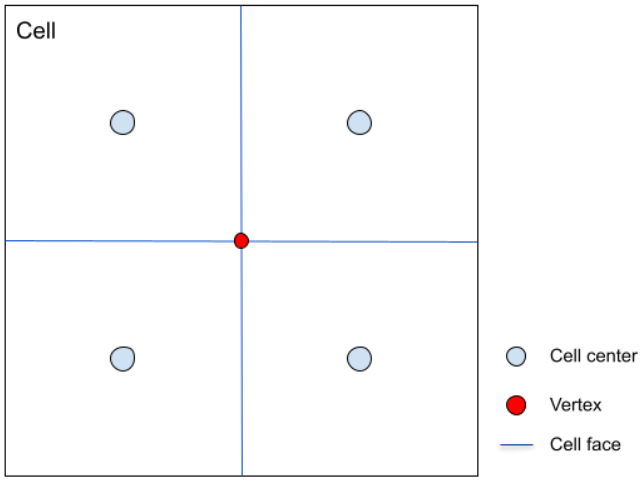


Figure 4.1: Sketch of how a domain can be split up into discrete control volumes with components related to the mesh.

The second step revolves around discretizing the equations so that they can be solved. This is done by first spatially discretizing by considering the integral form of the conservation equation on the steady state form, which can be seen for a general scalar ψ:

$$\int_{CS} \rho\psi\mathbf{u}\cdot\mathbf{n}\, dS = \int_{CV} \Gamma_\psi \nabla\psi\cdot\mathbf{n}\, dS + \int_{CV} m_\psi\, dV \tag{4.1}$$

where CS and CV denote the control surface and volume respectively, ρ is the density, $\mathbf{u}$ is the velocity, $\mathbf{n}$ is the normal vector out of surface S, Γ represents the diffusivity and m_ψ represents a source term within the control volume.

This equation applies to the entire domain, as well as each CV individually. The total conservation equation for the domain can therefore be found by summing the conservation equation for each CV, this is because the surface integrals across the inner faces cancel out. This can easily be visualized by looking at figure (4.2) where the surface flux on face e would be equal and opposite with the references of nodes P and E.

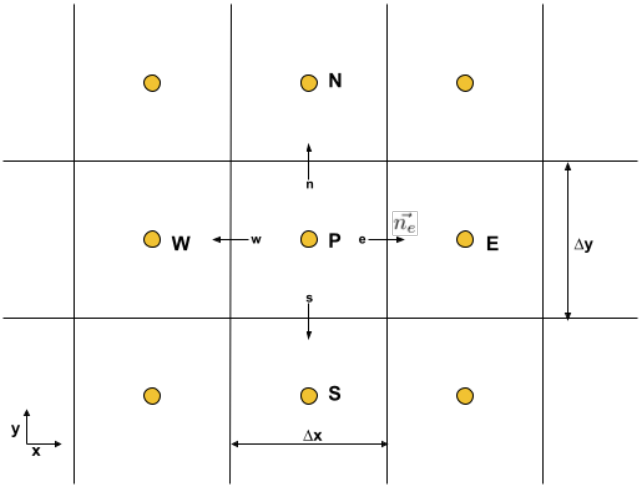


Figure 4.2: Illustration of a 2D CV domain. The yellow dots and capital letters indicate the computational nodes, faces are denoted by non-capitalized letters and $\vec{n_e}$ is an example of the vector out of face e. The figure is inspired by [38].

Since the total flux through the CV boundaries is equal to the sum of the integrals at each surface it can be expressed as:

$$\int_{CS} f\, dS = \sum_k \int_{CS_k} f\, dS \tag{4.2}$$

where f can either represent the convective ($\rho\psi\mathbf{u}\cdot\mathbf{n}$) or diffusive ($\Gamma\nabla\psi\cdot\mathbf{n}$) flux vector normal to the considered CV face. The values of the flux are not known across the entire face which means that further approximations are needed in order to compute the integrals. The simplest way of approximating the integral is by utilizing the midpoint rule where the integral is assumed to be equal to the product of the integrand and the face area.

$$F_e = \int_{CS_e} f\, dS = \bar{f}_e S_e \approx f_e S_e. \tag{4.3}$$

However, the value of f_e is not known in this equation as the needed values are stored in the centre of the CV at the computational nodes and not at the CV boundaries. It is therefore necessary to

utilize an interpolation scheme to get representative values at the face. One of the simplest ways to approximate this value is by utilizing the central differencing scheme (CDS) which uses linear interpolation between the two adjacent nodes:

$$\psi_e = \psi_E \lambda_e + \psi_P (1 - \lambda_e) \tag{4.4}$$

where the subscripts denote the locations described in figure (4.2) and λ_e denotes a linear interpolation factor defined as:

$$\lambda_e = \frac{x_e - x_P}{x_E - x_P}. \tag{4.5}$$

The linear profile of the CDS scheme allows for a simple approximation of the gradient at the face, given by:

$$\frac{\partial \psi}{\partial x} = \frac{\psi_E - \psi_P}{x_E - x_P}. \tag{4.6}$$

There exist many more interpolation schemes of varying complexity and order of accuracy, some examples of these are the upwind difference scheme (UDS), quadratic upwind interpolation (QUICK), as well as other higher order schemes. An in depth discussion of some select schemes can be found in the book by Ferziger [38].

The source term requires integration over the entire volume of the CV. This can be approximated by taking the mean value of the source term, which can be further assumed, as the value at the node, and multiplying it with the CV volume:

$$M_{\psi,P} = \int_{CV} m_\psi dV \approx m_{\psi,P} \Delta V \tag{4.7}$$

where $m_{\psi,P}$ is the value of the source term at the P node, in the volume centre. This solution becomes exact if the source term is constant, however, it does vary and depend on other variables for many applications. For such instances, it is typical to approximate it as a linear function

$$m_{\psi,P} \Delta V = m_u + m_P \psi_P. \tag{4.8}$$

It is now possible to substitute the approximations described in equations 4.4, 4.6 and 4.8 back into the general conservation equation which yields the algebraic equation for a given CV.

$$a_P \psi_P + \sum_k a_k \psi_k = M_{\psi,P} \tag{4.9}$$

where k denotes all adjacent faces to the computational node P, and a depends on all the relevant values that are stored in the nodes, like density, velocity, diffusion etc. This equation can then be applied to every CV, which results in a system of linear equations that can be constructed as a matrix with the form:

$$\mathbf{A}\psi = M. \tag{4.10}$$

4.1.2 Linear solver

The continuous exact solution of the PDE differential equation has now been discretized into a set of linear algebraic equations with the form of equation 4.9, but it is still necessary to solve these

equations to find an approximation. The matrix that is constructed from the set of equations will be sparse for most systems, meaning that most values will be 0, figure 4.3 shows the equations matrix for a discretization of 5x5 nodes. Each row in the figure represents equation 4.9 for any CV utilizing the CDS scheme.

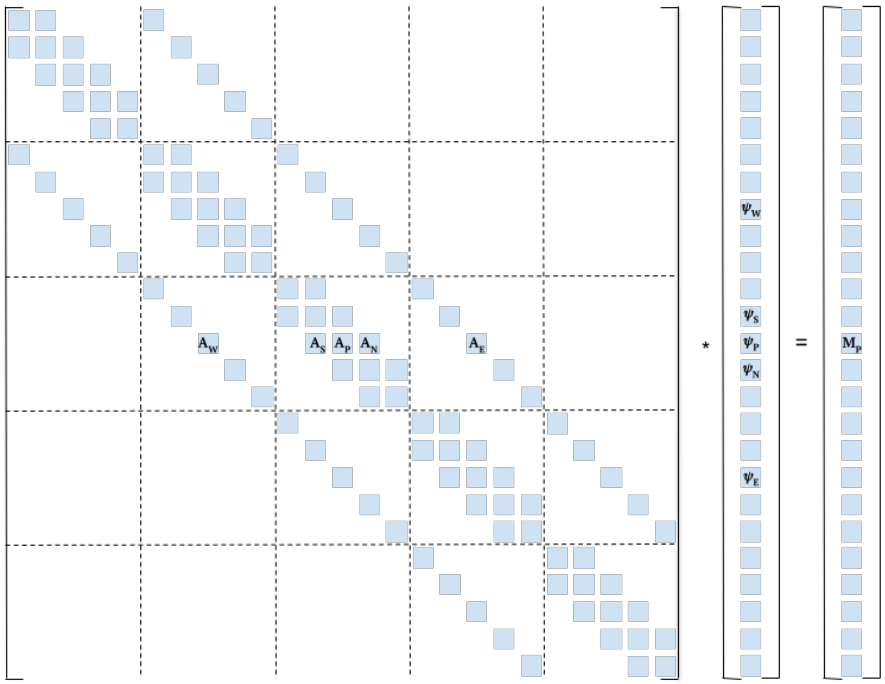


Figure 4.3: Matrix structure for a 5x5 computational domain. Shaded boxes represent non-zero indices with non-zero values. Each row represents the algebraic equation for a CV. Figure inspired by [38].

There exist many different methods for solving the set of equations, both direct and iterative. Direct methods such as the Gauss elimination method are very simple and accurate, they are however limited to sparse and linear systems. They are also very computationally demanding, where Gauss elimination needs $\frac{n^3}{3}$ number of operations for a system containing n amount of equations. For CFD simulations that can contain millions of equations it is simple to understand how this can be computationally unfeasible. For this reason, in addition to being able to solve non-linear systems, iterative methods are used.

The general concept of an iterative method is to guess an initial value for the solution, and utilizing the equations in an iterative manner that gradually improves the approximation over time. Looking at an arbitrary iteration n that has not converged yet for equation 4.10:

$$A\psi^n = M - \rho^n, \tag{4.11}$$

where ψ^n represents the current approximation in step n, and ρ^n is a residual, which represents the difference between the actual value of M and the value of M^n that is calculated from $\mathrm{A}\psi^n$. Subtracting this from equation 4.10 yields:

$$A(\psi - \psi^n) = A\epsilon^n = \rho^n \tag{4.12}$$

where ϵ^n is the iteration error given by the difference between the converged and current solutions of ψ. The goal of an iteration is to reduce the size of the residual, which also reduces the iteration error. It follows from the equation that as ρ^n goes to 0 the approximation converges. There exist many different algorithms and methods for determining the next iteration value, ψ^{n+1}, such as

the Jacobian and Gauss-Seidel methods for linear equations and conjugate gradient solutions like the CGS and GMRES algorithms for non-linear equations, all of these methods are explained by Ferziger and can be found in his book [38].

4.1.3 Time advancement

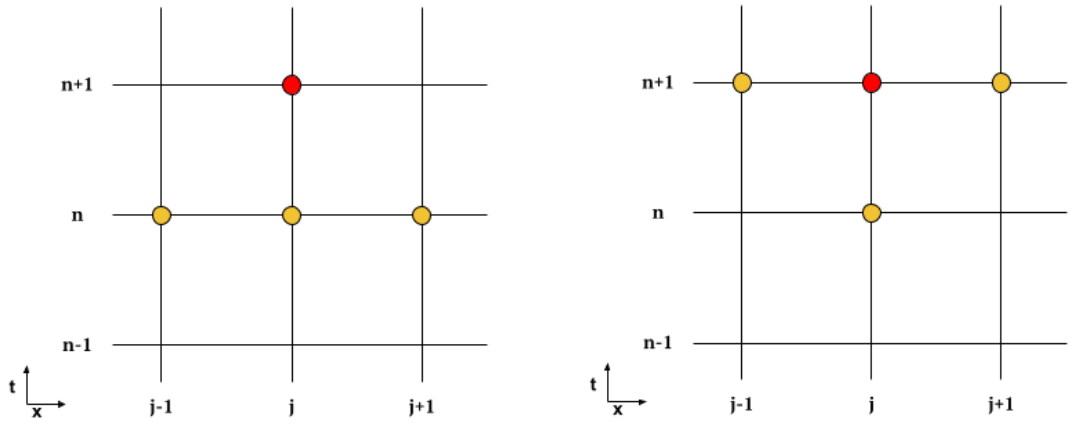
The governing equations for transient simulations need to be discretized in both space and time. The spatial discretization is done in the exact same way as for steady state, as was previously discussed. Discretization of the temporal part is done similarly by creating a time coordinate system in which the integral of the transient term is evaluated along [39]. The general formula for time evolution of variable ψ is given by:

$$\frac{\partial(\rho\psi)}{\partial t} = F(\psi) \tag{4.13}$$

where the left hand side of the equation represents the temporal term and the function on the right hand side represents the spatial terms i.e. convection, diffusion and sources at a given time step. A first order discretization of the time derivative in equation 4.13 can be written as:

$$\frac{\psi^{n+1} - \psi^n}{\Delta t} = F(\psi) \tag{4.14}$$

where Δt is the temporal discretization and the superscripts, n and n+1, represent which time step is being evaluated. Temporal discretization methods can broadly be split into two categories, implicit and explicit. A simple numerical stencil example can be seen for both in figure (4.4).



(a) Explicit Euler numerical stencil.

(b) Implicit Euler numerical stencil.

Figure 4.4: Numerical stencils for Explicit and Implicit Euler schemes, where central differencing schemes are used for the spatial discretization (x-axis). The yellow circles indicate which nodal values are being utilized for the calculations. The red circle represents the node that is currently being calculated.

Explicit discretization methods use the values of F(ψ) at the current time step to calculate the value in the next time step, as can be seen in figure (4.4a). This means that there is only one unknown variable, ψ^{n+1} and these methods are relatively simple to solve. One of the main drawbacks of these types of methods is that they are only conditionally stable and it is therefore imperative that care is taken when choosing time step size, as oscillations and divergence can easily occur if one is not careful.

The numerical stencil for an implicit scheme can be seen in figure (4.4b). It can be seen from the figure that all fluxes and source terms are being evaluated in the new time step. This means that both the left and right sides of equation 4.13 will contain unknown values. This leads to a set of linear algebraic equations quite similar to those found for the steady problem, which means that they can be solved iteratively. The implicit schemes are unconditionally stable, which means that an arbitrarily large time step can be chosen, care should however still be taken as large time steps still produce oscillatory results [38].

4.1.4 Meshing

The process of splitting the solution geometry into a grid of smaller volumes is typically called grid generation or meshing. The first step is creating a model of the real world geometry where the fluid flow takes place. This geometry is then split into a grid of non-overlapping control volumes/cells that mirrors the real world model, where physical boundaries coincide with the CV boundary conditions [40].

The choice of grid size i.e. how many CVs are used to represent the model is of great importance in CFD. A large grid size will lead to a coarse mesh with few nodes and a less accurate approximation while a small grid size will lead to a fine mesh where a lot of computation is needed [38].

The choice of the CV shape can also be of great importance depending on the complexity of the geometry being modelled. While simple geometries like a rectangular pipe can easily and accurately be modelled by using cubic CVs, this might not be the case for models with round edges or similar phenomena. There exist many different methods to create an accurate computational domain for such models which are explained in detail in the book by Versteeg [41].

4.1.5 Error and uncertainty

Representing a continuous system by discretization necessarily introduces a number of different types of error, including but not limited to; various numerical errors [36] as well as user errors and model uncertainties. It is important to recognize and know the sources and magnitude of these errors so that they can be reduced and thus a useful result may be obtained.

One of the more subtle types of errors, but with large ramifications, are user and software errors. What is meant by this is wrongful implementation and or use of the code. Simple errors like inputting a digit too many or too few for any given parameter can be an easy mistake to make but have serious impacts on the final solution and waste computational resources. More systematic errors can stem from how the numerical model has been implemented within the source code of the software. Errors such as these can be avoided through adequate training of the user and systematic quality assurance by the developer [41].

The numerical errors in a CFD simulation can be split into three main categories: round off errors, iterative convergence errors and discretization errors. Round off errors arise from the way real numbers are represented by floating points in computers. Certain arithmetic operations, like subtraction of two almost identical large numbers, with floating points can cause significant rounding errors that can propagate through the entire system. It is therefore necessary to consider how and which real numbers are used for computations so the rounding error stays minimal [42].

The iteration error, which was briefly shown in equation 4.12, comes from the difference between the current approximated solution and the converged solution. This means that the iteration error can, in theory, be close to 0 if every iterative process is allowed to converge. This requires a lot of computational resources however, and the iterative processes are therefore typically terminated when the size of the iteration error is sufficiently small and has reached a predetermined tolerance value. A common way of determining the current iteration error is by utilizing the residual ρ. The absolute values of the residuals for every CV after k iterations are added together, which creates the global residual:

$$(\hat{\rho}^{\psi})^k = \sum_{i=1}^{M} |(\rho_i^{\psi})^k| \tag{4.15}$$

where $\hat{\rho}^{\psi}$ and ρ_i^{ψ} are the global and local residuals for an arbitrary variable ψ respectively. The absolute value is taken to avoid cancellations of residuals of opposite values. A normalisation factor is typically applied to the global residual as different flow conditions could lead to largely different acceptable residual sizes.

$$(\hat{\rho}^{\psi})_N^k = \frac{(\hat{\rho}^{\psi})^k}{\hat{F}_{\rho\psi}}. \tag{4.16}$$

There are many different ways of deciding the normalization factor, one of them being:

$$\hat{F}_{\rho\psi} = (\hat{\rho}^{\psi})^{k_0}, \quad 2 \leq k_0 < 10 \tag{4.17}$$

where it is determined to be the global residual after just a few iterations. Typically before 10 iterations have occurred, but more than 1. This allows the normalization factor to stabilize at a reasonable, but still large enough value to be usable. This normalization factor then allows for the same residual tolerance to be set for all cases regardless of the flow conditions. The default tolerance values that are supplied in CFD software have typically been rigorously tested by developers to give acceptable results, however, could be decreased further if the solution accuracy is being tested. The iterative process is said to have **converged** when the residuals for all transport properties fall below the defined tolerance [41].

The discretization errors are caused by the way the continuous mathematical problem is approximated and refers to the difference between the converged solution and the exact one, or in other terms, how **accurate** it is. Two of the most important factors when attempting to decrease discretization error is the grid/mesh size and choice of interpolation scheme. This is due to the fact that a higher order interpolation scheme will lead to a more rapid decrease in discretization error as the mesh is refined due to a reduction in the truncation error of the assumption. Mesh and time step refinement inherently leads to reduction in discretization error as the computational domain more accurately approximates the continuous domain. However, this rapidly increases the computational demand and a compromise between refinement and computational resources needs to be made [41].

Uncertainties in a CFD simulation differ from errors in that they stem from a lack of knowledge and can be split into two main groups; input and model uncertainties. Input uncertainties are caused by limited data and variations in the real world. An idealized model might deviate from the real life counterpart due to production tolerances or similar phenomena. Missing information of local fluid property variations or uneven friction across a boundary are both examples of uncertainties that will impact the flow but are impossible to have intimate knowledge of when attempting to do simulations [41].

Model uncertainties can arise when a sub-model is utilized for a flow that is more complex than the one it was developed and validated for, meaning that it is assumed that the model can still be applied and be valid, while this might not be true in reality. The model could also be based on empirical data, which itself contains uncertainties. And lastly simplifying assumptions such as simulating in 2D or utilizing incompressible flow could introduce significant uncertainty in the results [41].

4.1.6 Solution algorithms

By looking at the continuity equation (3.8) and momentum equation (3.15, 3.18) it can be seen that the equations are coupled for velocity, however, the pressure only appears in the momentum equation and there is no apparent transport equation for it. This is an issue when dealing with

incompressible flows, as the pressure is not able to be linked to density and solved that way. This leads to a constraint on the solution of the flow field, where the correct pressure field has to be applied to the momentum equation to satisfy continuity. The problem of coupling between the pressure and velocity can be solved by utilizing an iterative solution strategy, like the SIMPLE or PISO algorithms where the initial pressure and velocity fields are guessed and then iteratively improved until they converge [41].

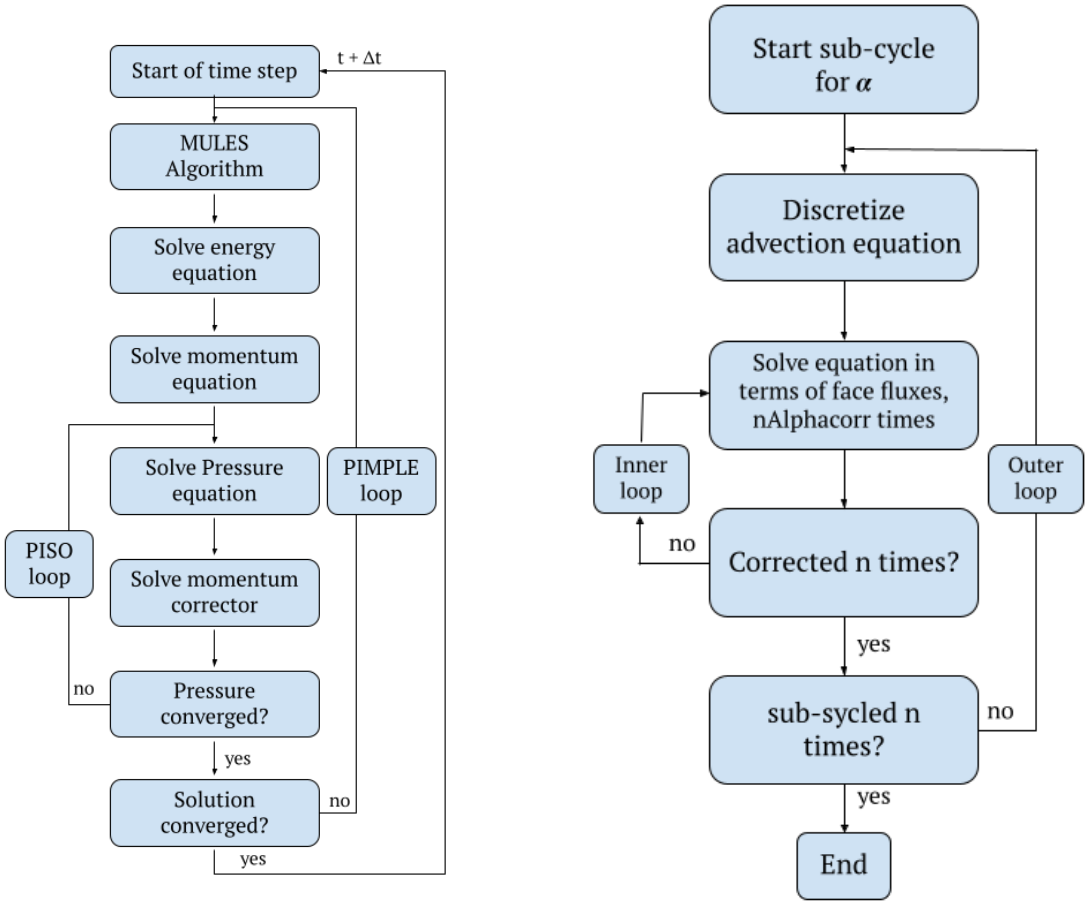
There are however some issues that need to be addressed when attempting to extend such algorithms, which were developed for singular flows, to multiphase problems. Spalding [43] provides three issues that need to be accounted for when attempting to do so. The first being that there are more field variables that need to be determined, thus increasing the computational demand. The second being that multiphase flows introduce more than one continuity equation. This means that there is no single unique way to determine the pressure correction to ensure continuity satisfaction. And lastly that the momentum exchange term between the multiple phases needs to be fully implicit due to the strong coupling it has with the momentum equations. Syamlal [44] outlines two additional problems with this in relation to granular flows. Firstly the handling of the packed region at minimum and maximum solid packing. And secondly, the difficulty in calculating field variables, the velocity in particular, when the phase fraction goes to 0, as they are not defined.

This thesis studies multiphase flow phenomena with the use of two different CFD software: MFiX [45] and OpenFOAM [46]. These softwares handle the coupling of multiphase velocities and pressure differently, and a brief overview of the solution algorithms used in both will be presented here.

The MFiX two-fluid model (TFM) utilizes a staggered grid, meaning that the velocities are stored and discretized at the CV faces, and not in the centre together with the pressure. This is done to prevent the phenomenon of checkerboard pressure fields, which can occur for collocated grids. It calculates all equations for the velocity components at the same time, this is to ensure that the momentum exchange terms are implicit. This however leads to an unstructured solution matrix that is then remedied by partially decoupling the exchange terms. The closely packed solid volume fraction is handled by deriving a state equation that relates the solid volume to the solid pressure, which limits the maximum packing to an appropriate value. The field variables at the interfaces are handled by approximation, which increases stability [44]. The general solution algorithm in MFiX as described by Syamlal [44] can be presented as:

1. Time step starting point. Physical properties, exchange coefficients and reaction rates are calculated.
2. Velocity field is calculated based on the current pressure field: $\mathbf{u}_m^*$.
3. Calculate fluid pressure correction P_g' and update fluid pressure field while applying an under relaxation $\mathrm{P}_g = \mathrm{P}_g^* + \omega_{pg}\mathrm{P}_g$'.
4. Calculate velocity correction $\mathbf{u}_m$' for all phases, based on the fluid pressure correction and update the velocity fields $\mathbf{u}_m = \mathbf{u}_m^* + \mathbf{u}_m$'.
5. Calculate solid pressure gradient $\frac{\partial P_m}{\partial \alpha_m}$ and calculate the solids volume correction α_m'.
6. Update solids volume fractions $\alpha_m = \alpha_m^* + \omega_{ps}\alpha_m$', where under relaxation is only applied close to minimum and maximum packing.
7. Calculate velocity corrections and update velocity fields for all solid phases.
8. Calculate void fraction $\alpha_g = 1 - \sum_{m \neq g}^{M} \alpha_m$.
9. Calculate solids pressure from the state equation $\mathrm{P}_m = \mathrm{P}_m(\alpha_m)$.
10. Calculate temperatures and species mass fractions if applicable.
11. Use residuals calculated in steps 2, 3, 6, and 10 to check for convergence. If the convergence criteria are satisfied go to step 1 and start the next time step, if not go to step 2.

OpenFOAM uses the multiphaseEulerFoam solver to solve the multiphase problem by combining an Eulerian-Eulerian model with interface sharpening from the volume of fluid (VOF) method [47]. This solver utilizes two main algorithms, MULES and PIMPLE. The general overview of MULES (Multidimensional Universal Limiter with Explicit Solution), which serves as a corrector on the phase within a time step, and acts as a sub-algorithm within the PIMPLE algorithm can be seen in figure (4.5b). The PIMPLE algorithm combines the momentum correction equation from PISO and under relaxation from SIMPLE, in addition to allowing for several iterations within one time step [48]. PIMPLE also allows for larger time steps due to it increasing the stability of the solution, however a sufficiently small time step is utilized in the present work, making this feature inapplicable. An overview of PIMPLE can be seen in figure (4.5a).



(a) PIMPLE algorithm, inspired by [48].

(b) MULES algorithm, inspired by [47].

Figure 4.5: Solution algorithms utilized by the multiphaseEulerFoam numerical solver.

4.2 OpenFOAM

OpenFOAM (Open source Field Operation and Manipulation) is a free and open source CFD software written in the C++ programming language. It is distributed under the general public license, meaning that all users are free to alter and distribute it as they please. It began development in 1989 by Henry Weller under the name of FOAM, and was released to the public as OpenFOAM in 2014 [49]. OpenFOAM has received continuous development and is currently in its 10th version. It functions as a toolbox for letting any user develop customized numerical solvers that are able to solve any problem within continuum mechanics.

4.2.1 Case setup

The general case setup explained here is based on the tutorial case "cavity", which can be found at the location $FOAM_TUTORIALS/incompressible/icoFoam/cavity/cavity within the OpenFOAM tutorial directory. All OpenFOAM cases are constructed within a main directory, with three sub-directories as can be seen in figure (4.6).

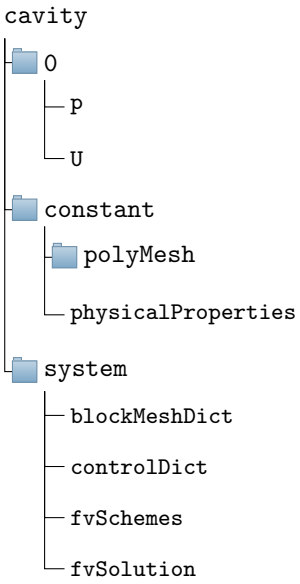


Figure 4.6: Directory structure for the OpenFOAM tutorial case, cavity.

The three sub-directories: 0, constant and system, that can be seen in figure 4.6, are standard in all OpenFOAM cases. The boundary conditions for all relevant variables are defined within the *0* directory, in this case for the pressure (p) and velocity (U). The initial conditions and fluid properties are stored and defined in the *constant* directory. And lastly, the *system* directory contains all the discretization and solution schemes that are utilized in the computations.

4.3 MFiX

MFiX (Multiphase Flow with Interphase eXchange) is an open source multiphase flow solver developed by the United States Department of Energy's national energy technology laboratory written in the Fortran programming language. MFiX was first released in the 1990s and has received continuous development, currently released as version 23.1.1. The MFiX solver comes packaged with many different multiphase models like for example MFiX-TFM (Two-Fluid Model) which is an Eulerian-Eularian model and MFiX-DEM (Discrete Element Method) which is an Eulerian-Lagrangian model. Many other more specialized models are also included in the software [50].

4.3.1 GUI

MFiX is released with a graphical user interface (GUI) which allows the user to quickly set up their desired simulation. The program opens to a screen where the user can select from many different presets and choose the one that most closely represents their simulation. Figure (4.7) shows how the MFiX GUI is present to the user after a preset is chosen.

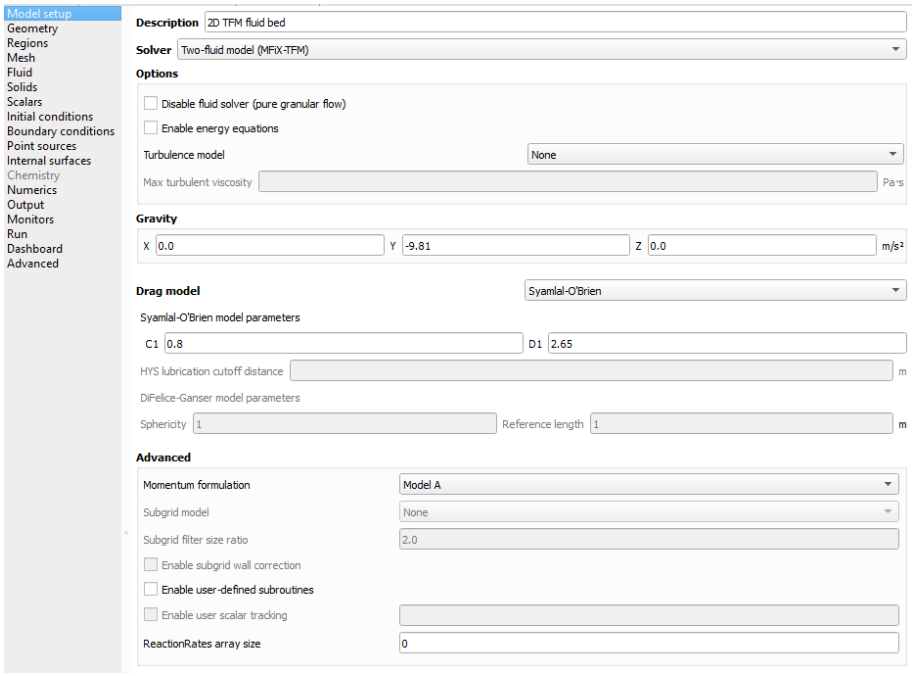


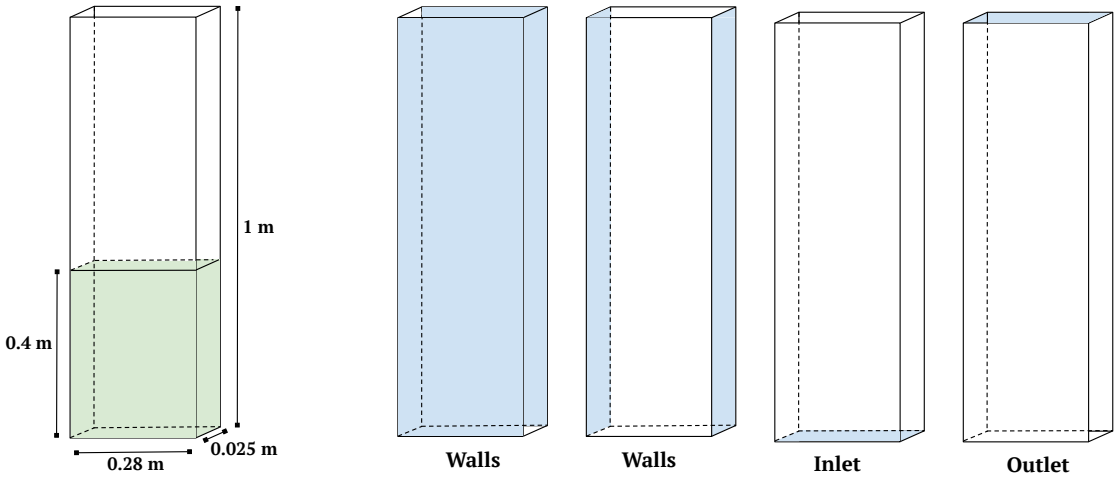
Figure 4.7: Portion of the MFiX GUI where model parameters are written. The 'model setup' is currently selected in this figure.

The menu on the left hand side of figure (4.7) can be clicked and allows for input of the different model parameters that are listed. The right hand side of the GUI, which changes depending on what menu is selected, is where the individual parameters are input.

5 Simulation setup

The simulations in this thesis are those of an Eulerian-Eulerian model of a fluidized bed in a thin column. Simulations are conducted both in 2D and 3D, where the Gidaspow and Syamlal O'Brien drag models are tested in a range of superficial gas velocities in both OpenFOAM and MFiX. The simulations copy the experimental setup by Taghipour [15] exactly. This is done for three main reasons, the first reason being that having experimental data for the simulations allows for validation of the results. Secondly, others have conducted similar simulations [18, 24], which can act as a point of reference for the expected accuracy of the results. And lastly, the simplicity of the experimental setup allows for testing of the fundamental properties of the flow without having to control for extra complexities.

The geometric setup of the model is made up of a rectangular column that is 0.28 m wide, 1 m tall and 0.025 m in depth. The bottom 0.4 meters of the geometry is filled with granular particles with a packing factor of 0.6, a sketch of the geometric setup can be seen in figure (5.1a). The grid size for all simulations was chosen as $\Delta x = 5mm$, since this was found to be optimal by Shi et al.[24]. This relates to a mesh of (56, 200, 1) for 2D simulations and (56, 200, 5) for 3D simulations.



(a) Geometry.

(b) Boundaries and their respective names.

Figure 5.1: Figure (a) shows the dimensions of the geometric setup used for simulations, where the light blue area represents the particle bed at the start of the simulation. Figure (b) shows the defined boundaries, and which names are used for them.

The relevant initial conditions are all set to either ambient values, or zero, as they have no effect and are not of interest, with pressure as the exception, in the present simulations and can be found in table 5.1.

Table 5.1: Initial conditions utilized in all simulations

Parameter	Value	Unit
Gas velocity	0	m/s
Particle velocity	0	m/s
Gas temperature	298	K
Particle temperature	298	K
Pressure	101325	Pa
Bed particle fraction	0.6	
Reactor particle fraction	0	

The flow in the simulations consists of two phases, a gas phase and a granular solid phase. The gas phase has the physical properties of air at ambient temperature and the solid phase has the

physical properties of the glass beads used in Taghipours experiment. The values used for all the physical properties for both phases can be found in table 5.2.

Table 5.2: Physical properties of the gas and solid phases used in the simulations, sourced from [15].

Gas properties	Value	Unit
Molar weight	28.9	g/mol
Viscosity	1.84e-05	Pa s
Prandtl number	0.7	
Temperature	298	K
Pressure	101325	Pa
Velocity	0.03-0.51	m/s
Particle properties	Value	Unit
Mean diameter	275	μm
Density	2500	kg/m^3
Maximum packing	0.63	
Initial packing	0.6	
Restitution coefficient	0.9	
Specularity coefficient	0.2	

The simulations have been conducted by testing a range of superficial gas velocities. The velocities that have been tested have ranged between half the minimum fluidization velocity and six times the minimum fluidization velocity. All the velocities that were tested were simulated with both the Syamlal-O'Brien and Gidaspow drag models. An assessment of the effect of the front and back walls were also conducted with 0.38 m/s superficial gas velocity with both drag models. All simulation cases can be seen listed in table 5.3 and were run in both OpenFOAM and MFiX.

Table 5.3: All simulation cases tested in both OpenFOAM and MFiX

Case	Gas velocity m/s	Drag model	Grid	Dimensions
1	0.03	Gidaspow	(56, 200, 1)	2D
2	0.1	Gidaspow	(56, 200, 1)	2D
3	0.2	Gidaspow	(56, 200, 1)	2D
4	0.38	Gidaspow	(56, 200, 1)	2D
5	0.46	Gidaspow	(56, 200, 1)	2D
6	0.51	Gidaspow	(56, 200, 1)	2D
7	0.03	Syamlal-O'Brien	(56, 200, 1)	2D
8	0.1	Syamlal-O'Brien	(56, 200, 1)	2D
9	0.2	Syamlal-O'Brien	(56, 200, 1)	2D
10	0.38	Syamlal-O'Brien	(56, 200, 1)	2D
11	0.46	Syamlal-O'Brien	(56, 200, 1)	2D
12	0.51	Syamlal-O'Brien	(56, 200, 1)	2D
13	0.38	Gidaspow	(56, 200, 5)	3D
14	0.38	Syamlal-O'Brien	(56, 200, 5)	3D

All simulations were run for 12 seconds of flowtime as this was found to be a good compromise between computational time and sufficient amount of data after the simulation had reached steady state.

5.1 Numerical settings in OpenFOAM

The OpenFOAM simulations are based on the fluidized bed tutorial case that can be found in the OpenFOAM tutorial directory at $FOAM_TUTORIALS/multiphase/multiphaseEulerFoam /laminar/fluidizedBed. The momentumumTransport.particles file was altered to use the same settings as those available in MFiX which can be seen in listing 1.

```
        granularViscosityModel      Syamlal; //Gidaspow
        granularConductivityModel   Syamlal; //Gidaspow
        granularPressureModel       Lun;
        frictionalStressModel       Schaeffer;
        radialModel                 CarnahanStarling;

        SchaefferCoeffs
        {
            Fr                      0.05;
            eta                     2;
            p                       5;
            phi                     30;
            alphaDeltaMin           0.05;
        }
```

Listing 1: Settings altered in momentumTransport.particles. The viscosity and conductivy are changed to mirror the current drag model being used.

All boundary conditions were kept the same between the tutorial case and these simulations, except for U.particles and Theta.particles, where the Johnson-Jackson boundary condition was applied to the walls to account for partial slip between the particles and the walls. How these conditions were applied in the code can be seen in listings 2 and 3. The inlet gas velocity was also changed in between simulations which can be seen in listing 4

```
    walls
    {
        type                    JohnsonJacksonParticleSlip;
        value                   $internalField;
        specularityCoefficient  0.2;
    }
```

Listing 2: Wall boundary condition in U.particles.

```
    walls
    {
        type                    JohnsonJacksonParticleTheta;
        value                   $internalField;
        restitutionCoefficient  0.9;
        specularityCoefficient  0.2;
    }
```

Listing 3: Wall boundary condition in Theta.particles.

```
    inlet
    {
        type                    interstitialInletVelocity;
        inletVelocity           (0 0.38 0); //0.03, 0.1, 0.2, 0.46, 0.51
        alpha                   alpha.air;
        value                   $internalField;
    }
```

Listing 4: Inlet gas velocity boundary condition in U.air

Table 5.4: Numerical schemes used in the OpenFOAM simulations, found in the fvSchemes file.

ddtSchemes	
default	Euler
gradSchemes	
default	Gauss linear
divSchemes	
default	none
"div\(phi,alpha.*\)"	Gauss vanLeer
"div\(phir,alpha.*\)"	Gauss vanLeer
"div\(alphaRhoPhi.*,U.*\)"	Gauss limitedLinearV 1
"div\(phi.*,U.*\)"	Gauss limitedLinearV 1
"div\(alphaRhoPhi.*,(h—e).*\)"	Gauss limitedLinear 1
"div\(alphaRhoPhi.*,K.*\)"	Gauss limitedLinear 1
"div\(alphaRhoPhi.*,\(p\—thermo:rho.*\)\)"	Gauss limitedLinear 1
div(alphaRhoPhi.particles,Theta.particles)	Gauss limitedLinear 1
"div\(alphaRhoPhi.*,(k—epsilon).*\)"	Gauss limitedLinear 1
div((((alpha.air*thermo:rho.air)*nuEff.air)*dev2(T(grad(U.air)))))	Gauss linear
vTau(U.particles)	Gauss linear
laplacianSchemes	
default	Gauss linear uncorrected
bounded	Gauss linear uncorrected
interpolationSchemes	
default	linear
snGradSchemes	
default	uncorrected
bounded	uncorrected
wallDist	
method	meshWave

Table 5.5: Solvers used in all OpenFOAM simulations.

Solvers	
"alpha.*"	
nAlphaCorr	1
nAlphaSubCycles	3
implicitPhasePressure	yes
smoothLimiter	0.1
solver	PBiCGStab
tolerance	1e-9
relTol	0
minIter	1
p_rgh	
solver	GAMG
smoother	DIC
tolerance	1e-8
relTol	0.0
p_rghFinal	
$p_rgh	
relTol	0
"U.*"	
solver	smoothSolver
smoother	symGaussSeidel
tolerance	1e-8
relTol	0
minIter	1
"(h|e).*"	
solver	smoothSolver
smoother	symGaussSeidel
tolerance	1e-8
relTol	0.0
minIter	1
maxIter	10
"Theta.*"	
solver	PBiCGStab
preconditioner	DILU
tolerance	1e-8
relTol	0
minIter	1
"(k|epsilon).*"	
solver	PBiCGStab
preconditioner	DILU
tolerance	1e-5
relTol	0
minIter	1
PIMPLE	
nOuterCorrectors	3
nCorrectors	2
nNonOrthogonalCorrectors	0
relaxationFactors	
equations	
".*"	1

The settings for the numerical schemes and solvers used in all OpenFOAM simulations can be found in tables 5.4 and 5.5 respectively.

5.2 Numerical settings in MFiX

The MFiX simulations are all based on the fluid_bed_2d 2D TFM fluid bed preset. The drag model is selected as either Syamlal or Gidaspow from the "Model Setup" as seen in figure 4.7. The geometric and physical parameters listed previously are entered into the subsequent menus seen in the same figure.

The viscous stress model, which can be found in the "TFM" sub-menu under "Solids" is changed to "Lun et al, 1984" which activates the frictional stress model which is then selected as Schaeffer. This process can be seen in figure (5.2).

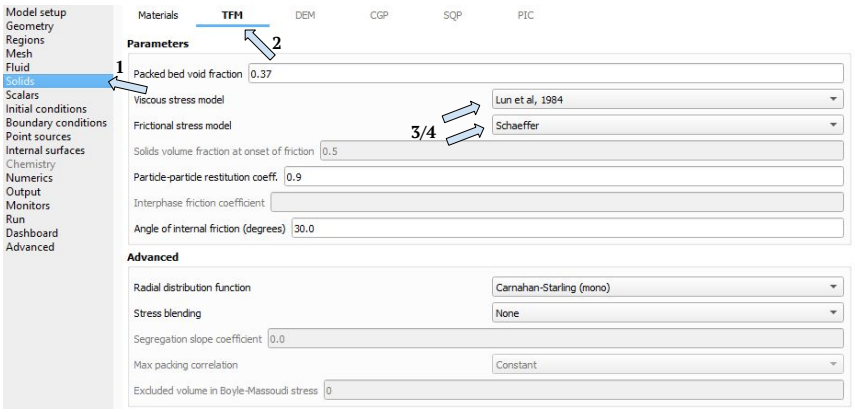


Figure 5.2: The process of activating the correct viscous stress model and frictional stress model in MFiX with the two fluid model

The choice of "Lun et al, 1984" also allows for the Johnson-Jackson boundary condition on the wall. This boundary condition is activated by setting the wall boundary type to "Mixed wall" and the solid wall type to "Partial slip". How the Johnson-Jackson boundary condition is activated in MFiX can be seen in figure 5.3, this process has to be repeated for each wall boundary.

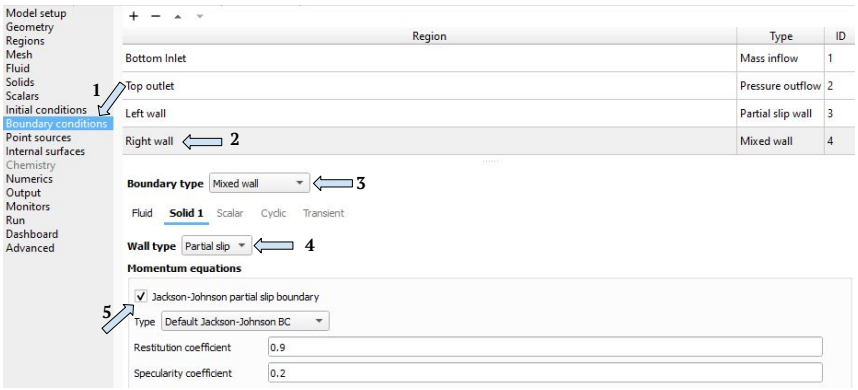


Figure 5.3: How the Johnson-Jackson boundary condition is applied to a wall boundary in MFiX, in this case the boundary named "Right wall".

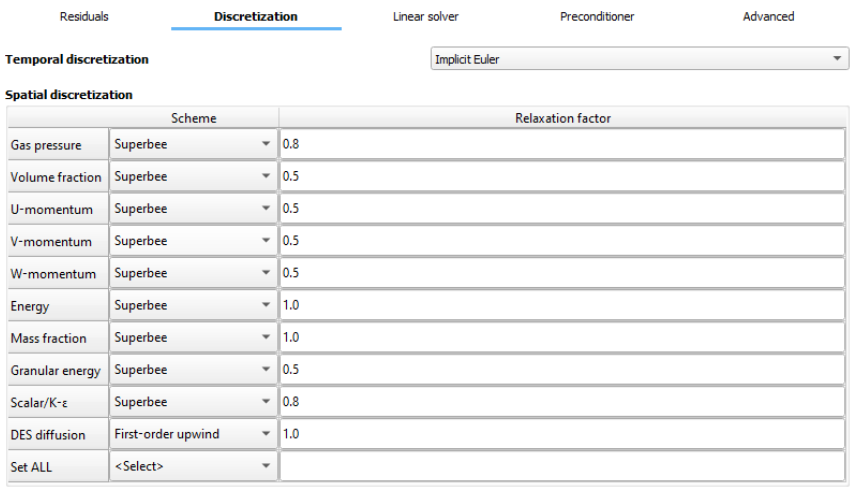


Figure 5.4: The discretization schemes used in the MFiX simulations

Residuals | Discretization | **Linear solver** | Preconditioner | Advanced

	Solver	Iter.	Tol.
Gas pressure	BiCGSTAB	20	0.0001
Volume fraction	BiCGSTAB	20	0.0001
U-momentum	BiCGSTAB	5	0.0001
V-momentum	BiCGSTAB	5	0.0001
W-momentum	BiCGSTAB	5	0.0001
Energy	BiCGSTAB	15	0.0001
Mass fraction	BiCGSTAB	15	0.0001
Granular energy	BiCGSTAB	15	0.0001
Scalar/K-ε	BiCGSTAB	15	0.0001
DES diffusion	BiCGSTAB	10	0.0001
Set ALL	<Select>		

Figure 5.5: The solvers used in the MFiX simulations

The numerical schemes, with relaxation factors, and solvers with their corresponding number of iterations, utilized in MFiX can be seen in figures (5.4) and (5.5) respectively.

6 Results

This section will present the results that were acquired from the different simulations. The results that will be presented first are the bed expansion and pressure drops found for all superficial gas velocities. After this, a more detailed presentation of the flow evolution taken at 0.4 seconds, 1 second, a snapshot of how the bed looks like after the bed has reached a steady fluidized state and a time average presentation of the volume fractions will be shown. Then the time averaged solid velocity and time averaged voidage will be presented at 0.4 meters bed height. This is done for superficial gas velocities of 0.1, 0.38 and 0.51 m/s, similar results for the other gas velocities can be found in Appendix A. The last results that will be presented are the flow evolution in the 3D simulations along with the time averaged volume fractions at the wall and the centre of the bed along the z-axis. This is then followed by a comparison between the time averaged solid velocities and voidages for the 2D and 3D simulations with the same superficial gas velocity.

6.1 Bed expansion

The height expansion in the bed was determined by plotting the time averaged solid fraction in the middle of the bed along the y-axis. The height where a threshold value of solid fraction lower than 1% was chosen to represent the bed expansion. This was done for all superficial gas velocities tested to see the change in bed expansion with inlet velocity. Figure (6.1) shows the bed expansions that were found with the Syamlal-O'Brien drag model in both MFiX and OpenFOAM. It can be seen from the figure that both softwares predict very little bed expansion for the lowest velocities and then increases linearly for velocities greater than 0.38 m/s. The bed expansion in the 3D cases is seen to be lower than for the same 2D cases. The figure also shows that OpenFOAM generally predicts a larger bed expansion than MFiX.

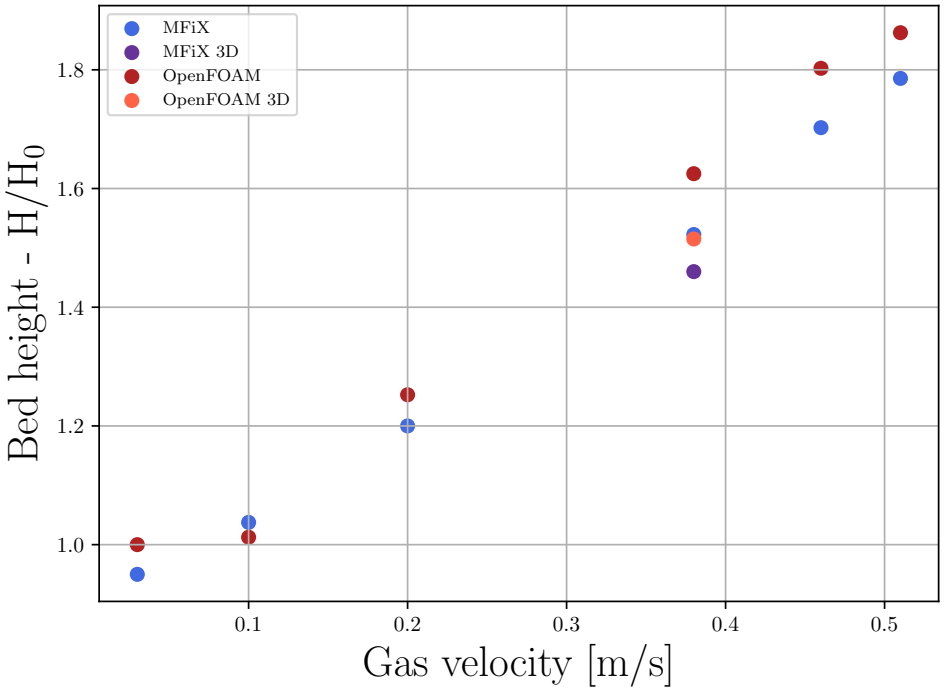


Figure 6.1: Bed expansion for different superficial gas velocities with the Syamlal-O'Brien drag model in both MFiX and OpenFOAM.

Figure (6.2) shows the bed expansions found with the Gidaspow drag model in MFiX and OpenFOAM. It can be seen from the figure that both softwares experience bed expansion for all velocities above the minimum fluidization velocity. The bed expansion looks to expand linearly with

the superficial gas velocity. Both 3D results predict significantly lower bed expansion than the 2D models. It can be seen when comparing with figure (6.1) that the bed expansion found with the Gidaspow model are greater than with Syamlal-O'Brien. The figure also shows that OpenFOAM generally predicts a larger bed expansion than MFiX for the Gidaspow drag model as well. Lastly, a bed contraction can be seen for the lowest gas velocity with both drag models in MFiX, but not OpenFOAM.

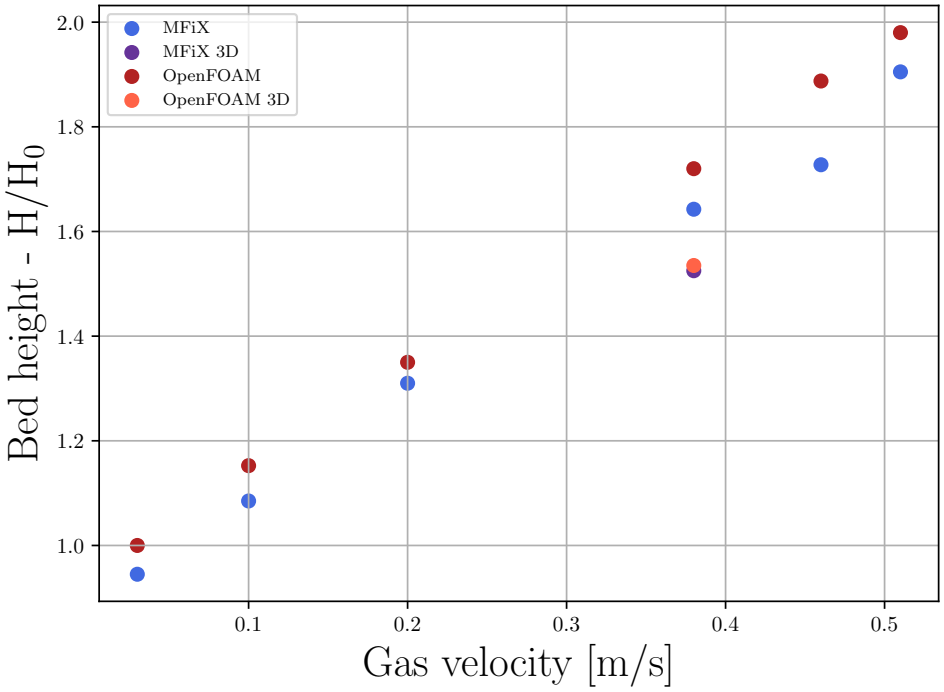


Figure 6.2: Bed expansion for different superficial gas velocities with the Gidaspow drag model in both MFiX and OpenFOAM.

6.2 Pressure drop

The pressure drop across the bed was determined by measuring the time averaged pressure at the outlet fluidized bed reactor at all superficial gas velocities tested. Both figures (6.3) and (6.4) show a trend where the pressure drop rapidly rises to a large value, and then slowly decreases as the superficial gas velocity is increased. It can also be seen that the pressure drop found with the Gidaspow model increases to it largest value for lower superficial gas velocities than Syamlal-O'Brien. It can be seen from both figures that MFiX predicts a higher pressure drop than OpenFOAM for all superficial gas velocities in both 2D and 3D. When comparing the 2D and 3D findings it can be seen that OpenFOAM predicts a lower pressure drop than 2D in 3D, while MFiX does the opposite.

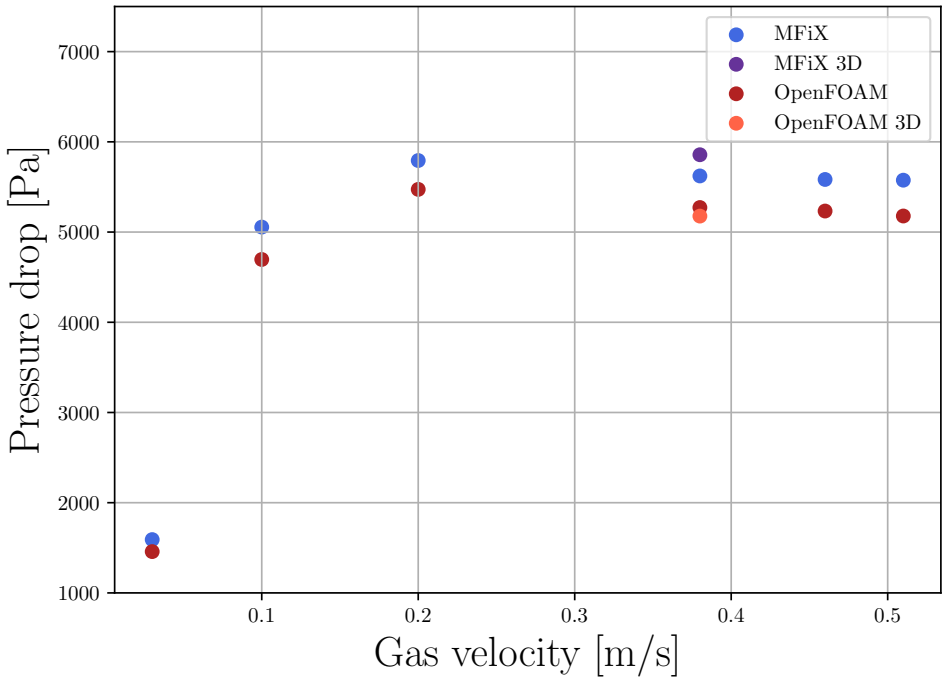


Figure 6.3: Pressure drop for different superficial gas velocities with the Syamlal-O'Brien drag model in both MFiX and OpenFOAM.

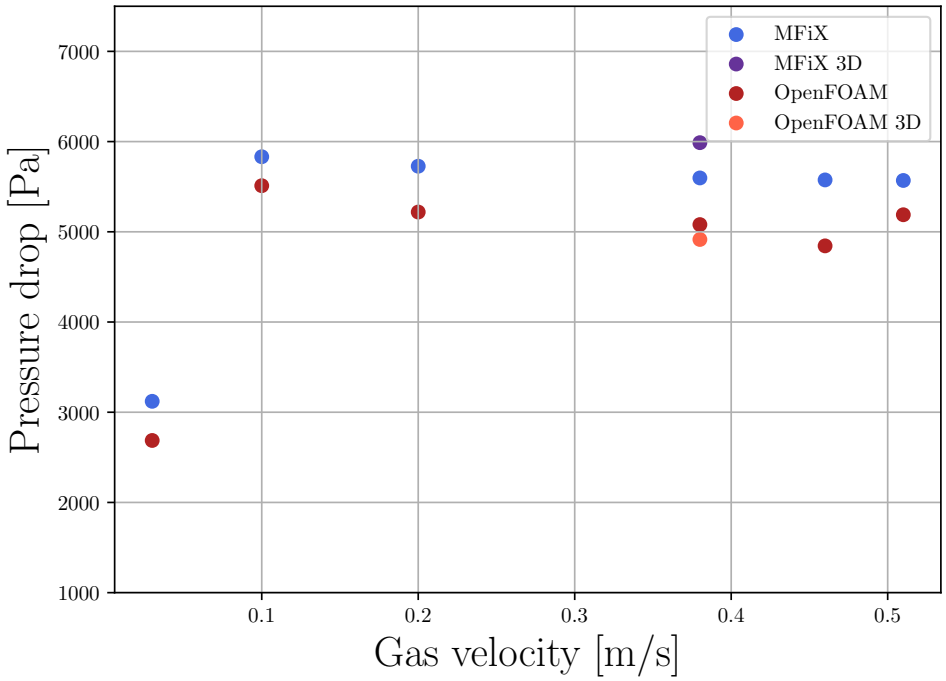


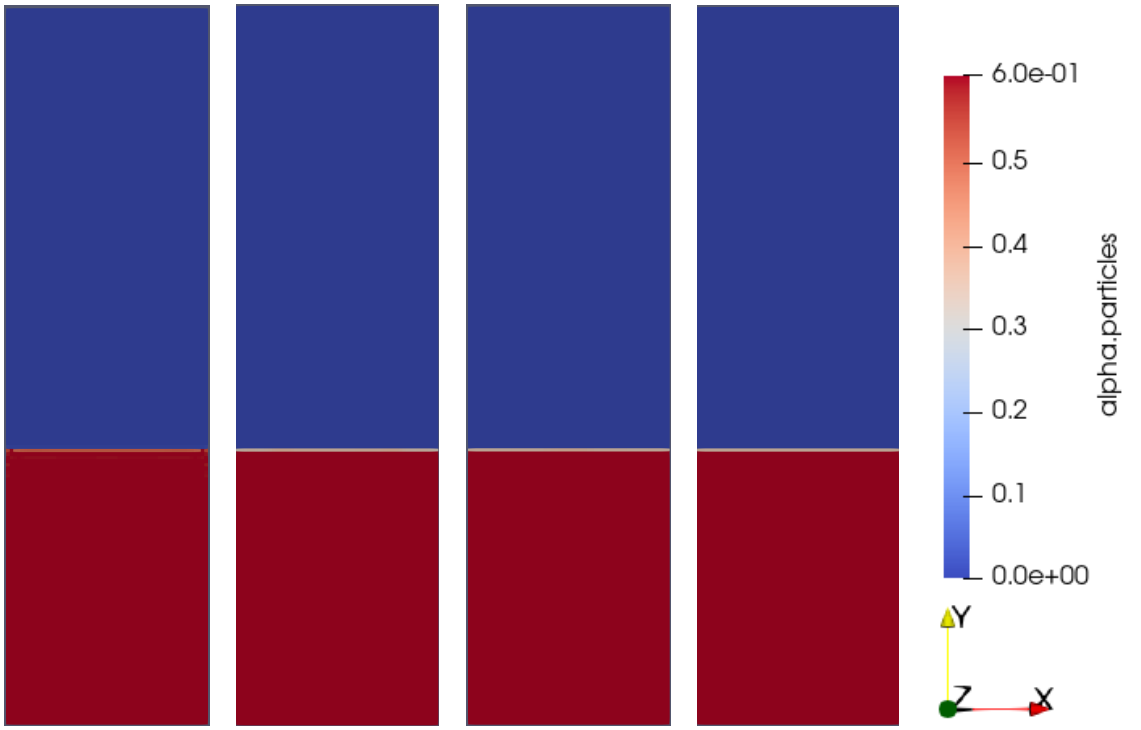
Figure 6.4: Pressure drop for different superficial gas velocities with the Gidaspow drag model in both MFiX and OpenFOAM.

6.3 Superficial gas velocity = 0.1 m/s

The evolution of the fluidized beds, as well as the steady state and time averaged solid volume fractions with a superficial gas velocity equal to 0.1 m/s can be seen in figures (6.5) and (6.6) for

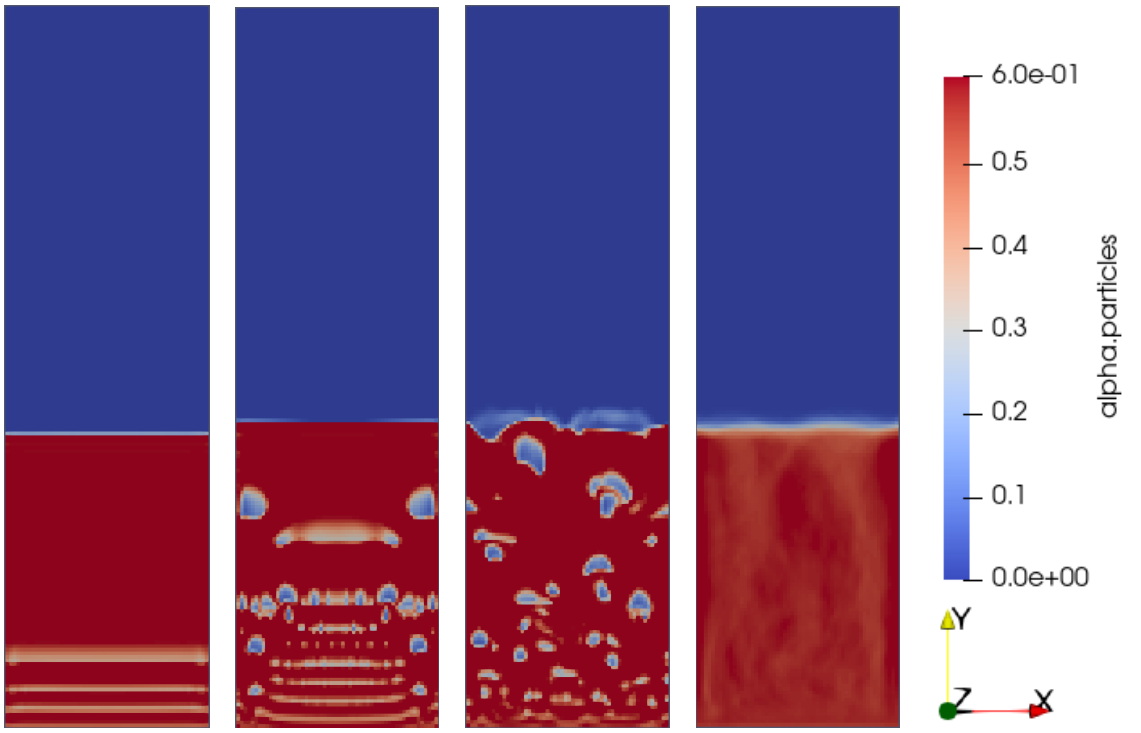
the Syamlal-O'Brien and Gidaspow drag models from MFiX, and figures (6.7) and (6.8) for the Syamlal-O'Brien and Gidaspow drag models from OpenFOAM. It can be seen from the figures that the simulations with the Syamlal-O'Brien model does not achieve fluidization in either software, even though the superficial gas velocity is above the minimum fluidization velocity.

The simulations utilizing the Gidaspow model can be seen to achieve fluidization, although the results from the different softwares are dissimilar. It can be seen in figures (6.6b) and (6.6c) that the results from MFiX create distinct bubbles consisting entirely of the gas phase, while the results from OpenFOAM seen in figures (6.8b) and (6.8c) show that the "bubbles" that are created consist of a mix of both the solid and gas phase.



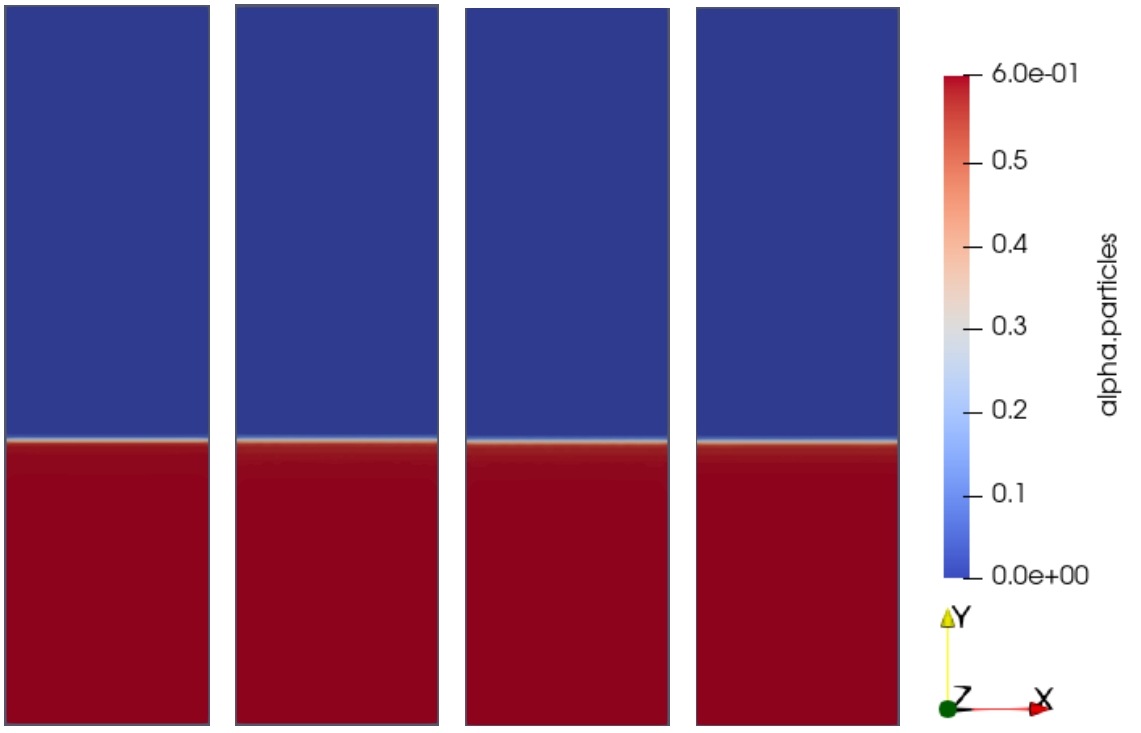
(a) 0.4 seconds (b) 1 second (c) Steady state (d) Time averaged (e) Scale bar

Figure 6.5: Particles distribution throughout the simulation with the Syamlal-O'Brien drag model with a superficial gas velocity equal to 0.1 m/s in MFiX.



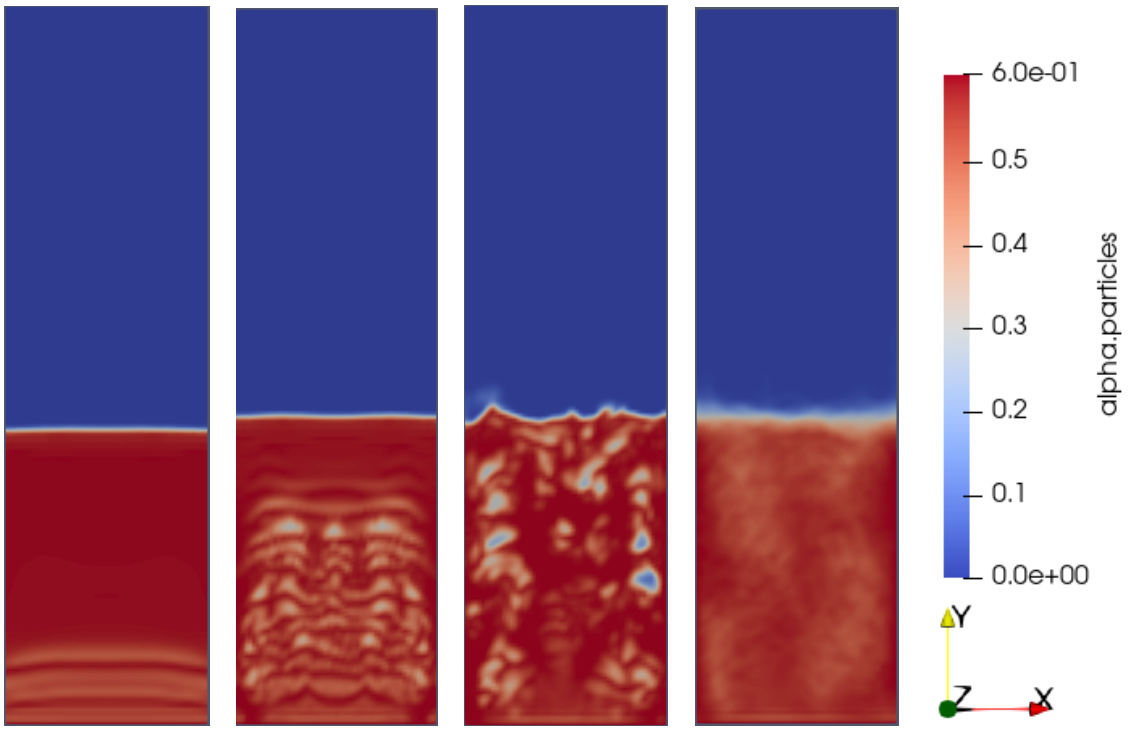
(a) 0.4 seconds (b) 1 second (c) Steady state (d) Time averaged (e) Scale bar

Figure 6.6: Particles distribution throughout the simulation with the Gidaspow drag model with a superficial gas velocity equal to 0.1 m/s in MFiX.



(a) 0.4 seconds (b) 1 second (c) Steady state (d) Time averaged (e) Scale bar

Figure 6.7: Particles distribution throughout the simulation with the Syamlal-O'Brien drag model with a superficial gas velocity equal to 0.1 m/s in OpenFOAM.



(a) 0.4 seconds (b) 1 second (c) Steady state (d) Time averaged (e) 0.4 seconds

Figure 6.8: Particles distribution throughout the simulation with the Gidaspow drag model with a superficial gas velocity equal to 0.1 m/s in OpenFOAM.

Figure 6.9 shows the time averaged solid velocity in the y-direction at 0.4 meters height with superficial gas velocity equal to 0.1 m/s for both drag models in both CFD softwares. It can be seen from the figure that there is no solid velocity from the Syamlal-O'Brien model in MFiX, but there is significant downward velocity across the entire bed from the same drag model in OpenFOAM. The lines for the Gidaspow model show that there is movement with both softwares, however, the results from OpenFOAM show a much larger gradient at the walls.

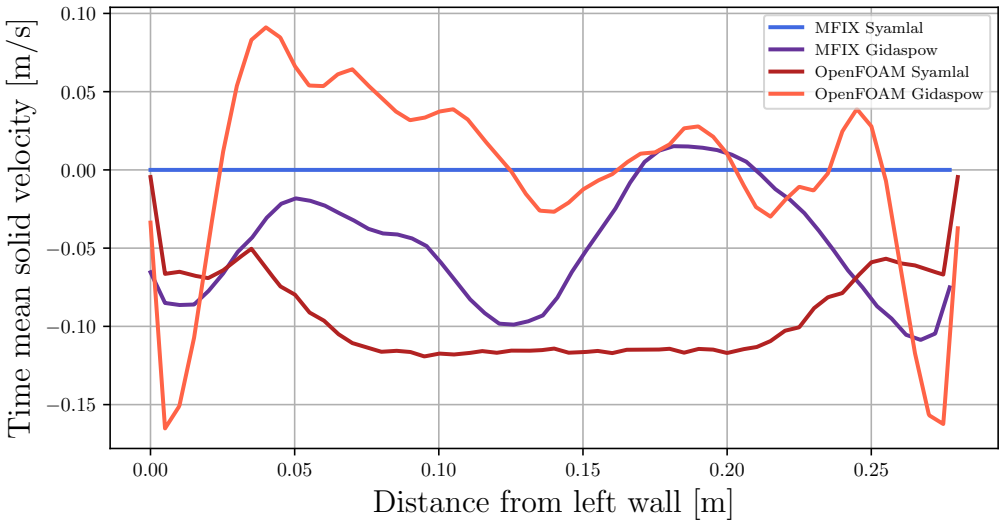


Figure 6.9: Time mean solid velocities measured in the range from 3 to 12 seconds with superficial gas velocity equal 0.1 m/s. Measurements taken at y = 0.4 m.

The time averaged voidage at y = 0.4 with superficial gas velocity equal to 0.1 m/s for the simula-

tions can be seen in figure (6.10), it shows that the average voidages predicted from OpenFOAM and MFiX are quite similar with the Gidaspow drag model, while there is deviation between the two softwares with Syamlal-O'Brien. MFiX predicts there to be no particles present, while OpenFOAM predicts a small fraction of particles, which shows that there is slight bed movement.

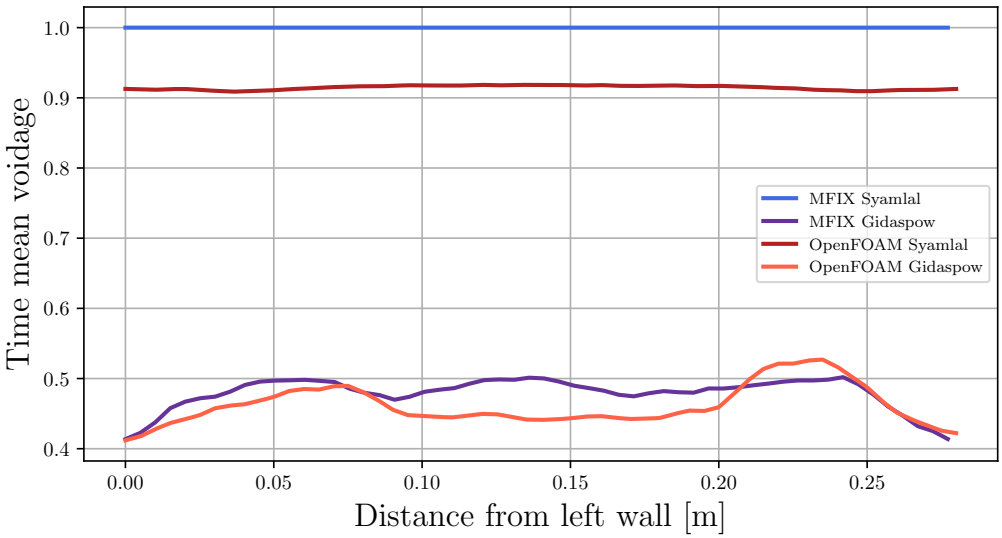
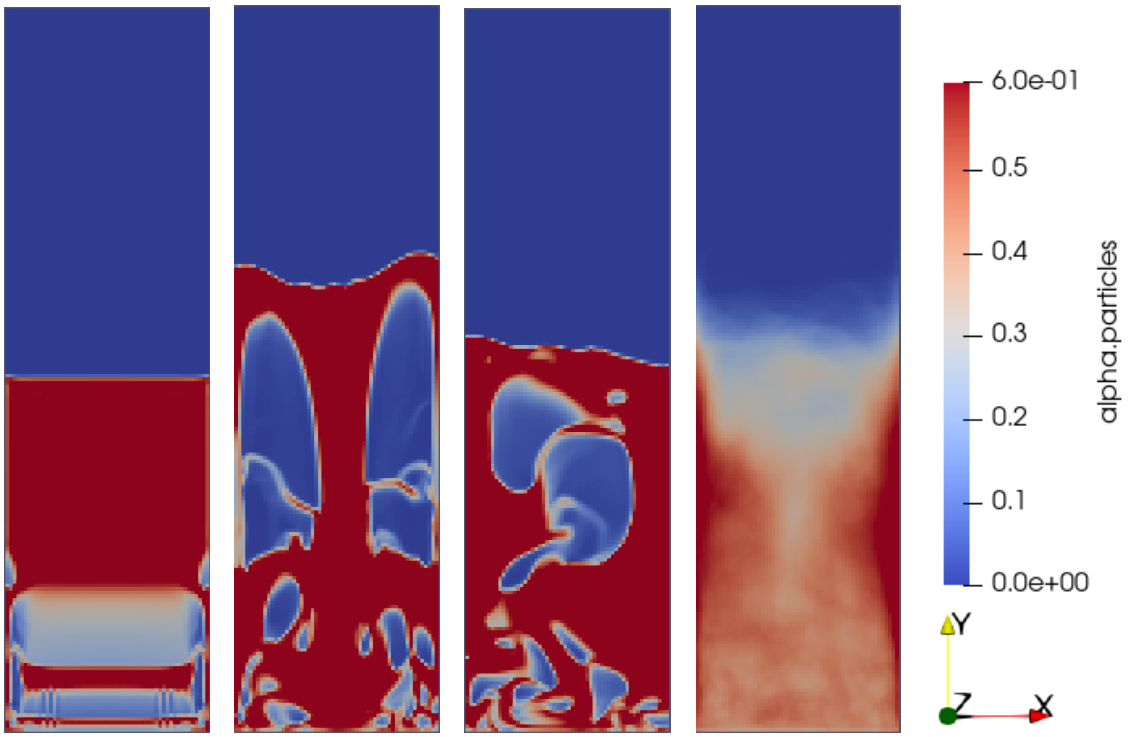


Figure 6.10: Time mean voidages measured in the range from 3 to 12 seconds with superficial gas velocity equal 0.1 m/s. The measurements are taken at y = 0.4 m.

6.4 Superficial gas velocity = 0.38 m/s

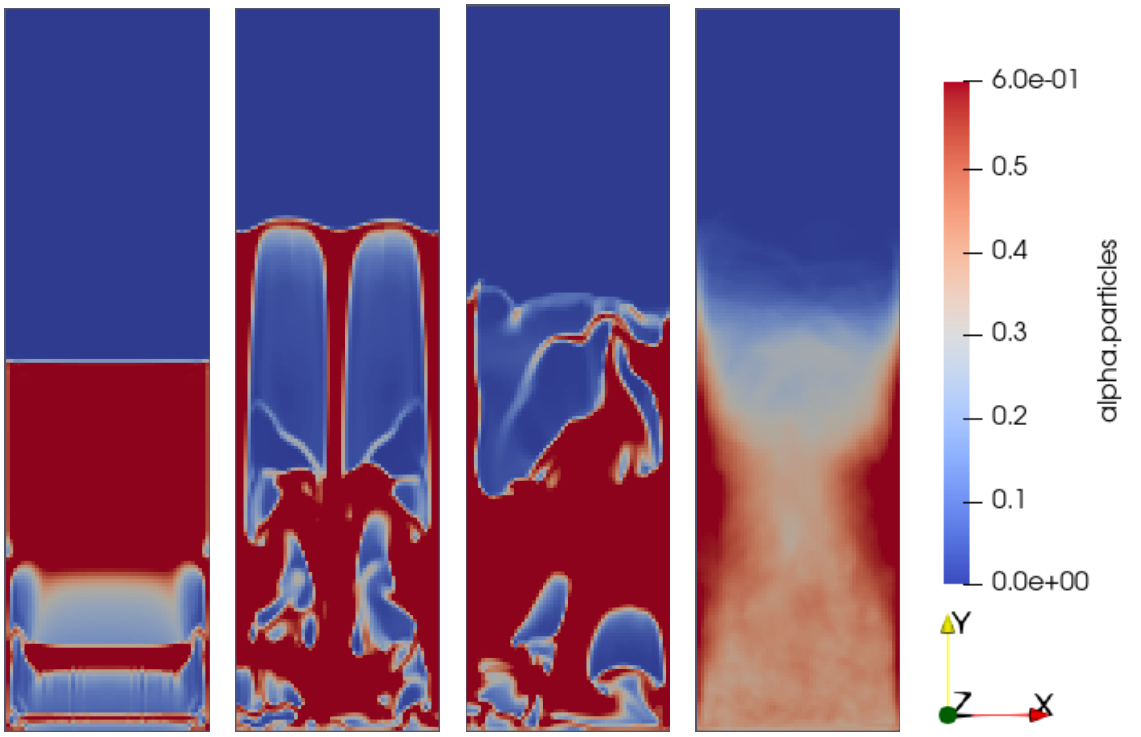
The evolution of the fluidized beds, as well as the steady state and time averaged solid volume fractions with a superficial gas velocity equal to 0.38 m/s can be seen in figures (6.11) and (6.12) for the Syamlal-O'Brien and Gidaspow drag models from MFiX, and figures (6.13) and (6.14) for the Syamlal-O'Brien and Gidaspow drag models from OpenFOAM. It can be seen from the figures that the Gidaspow drag model creates larger bubbles and has greater bed expansion than Syamlal-O'Brien, especially during the evolution of the fluidized state, at the 1 second mark, this is apparent in both MFiX and OpenFOAM. It can be seen in sub-figures (6.11d) and (6.12d) that MFiX has a larger region of time averaged particles that stretches out from the wall, and a lower bed apparent bed expansion comparatively to the time averaged particles in OpenFOAM, seen in sub-figures (6.13d) and (6.14d).

It can be seen from the figures that there seems to be a more distinct border between the phases in MFiX than in OpenFOAM, where the phases blend together at the interface, similar as to what could be seen in the results with superficial gas velocity equal to 0.1 m/s. This looks like it leads to more, but smaller bubbles in OpenFOAM, while there are just a few, substantially larger bubbles in MFiX.



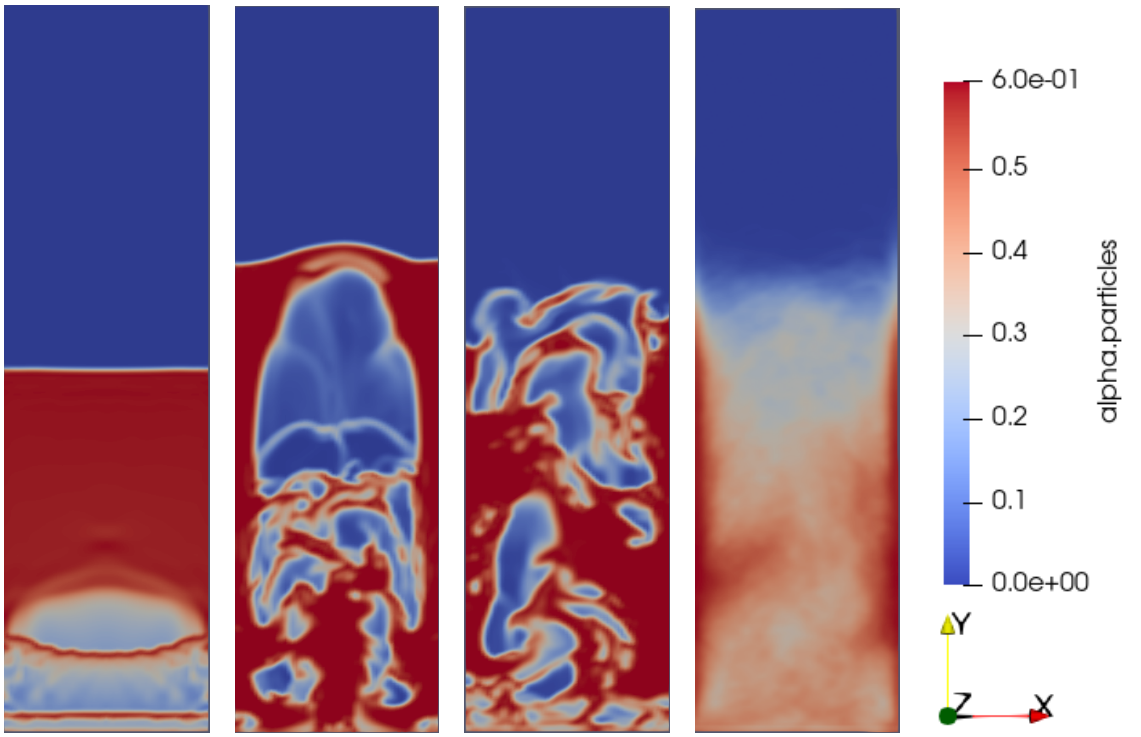
(a) 0.4 seconds (b) 1 second (c) Steady state (d) Time averaged (e) Scale bar

Figure 6.11: Particles distribution throughout the simulation with the Syamlal-O'Brien drag model with a superficial gas velocity equal to 0.38 m/s in MFiX.



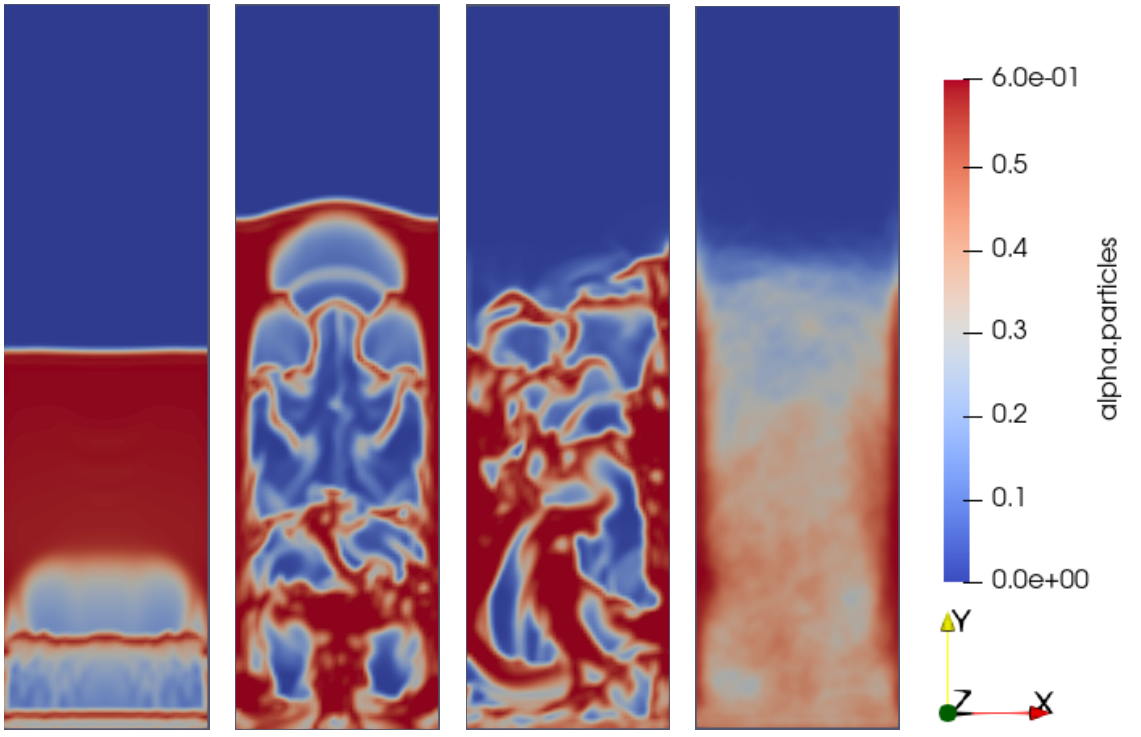
(a) 0.4 seconds (b) 1 second (c) Steady state (d) Time averaged (e) Scale bar

Figure 6.12: Particles distribution throughout the simulation with the Gidaspow drag model with a superficial gas velocity equal to 0.38 m/s in MFiX.



(a) 0.4 seconds (b) 1 second (c) Steady state (d) Time averaged (e) Scale bar

Figure 6.13: Particles distribution throughout the simulation with the Syamlal-O'Brien drag model with a superficial gas velocity equal to 0.38 m/s in OpenFOAM.



(a) 0.4 seconds (b) 1 second (c) Steady state (d) Time averaged (e) Scale bar

Figure 6.14: Particles distribution throughout the simulation with the Gidaspow drag model with a superficial gas velocity equal to 0.38 m/s in OpenFOAM.

Figure (6.15) shows the time averaged solid particle velocity in the y direction at 0.4 meters height with superficial gas velocity equal to 0.38 m/s for both drag models in both MFiX and OpenFOAM. It can be seen from the figure that all lines follow the same trend with downward velocity close to the walls and a parabolic upwards velocity around the centre of the bed. OpenFOAM does however predict significantly larger velocities than MFiX, especially close to the wall and close to the center of the reactor.

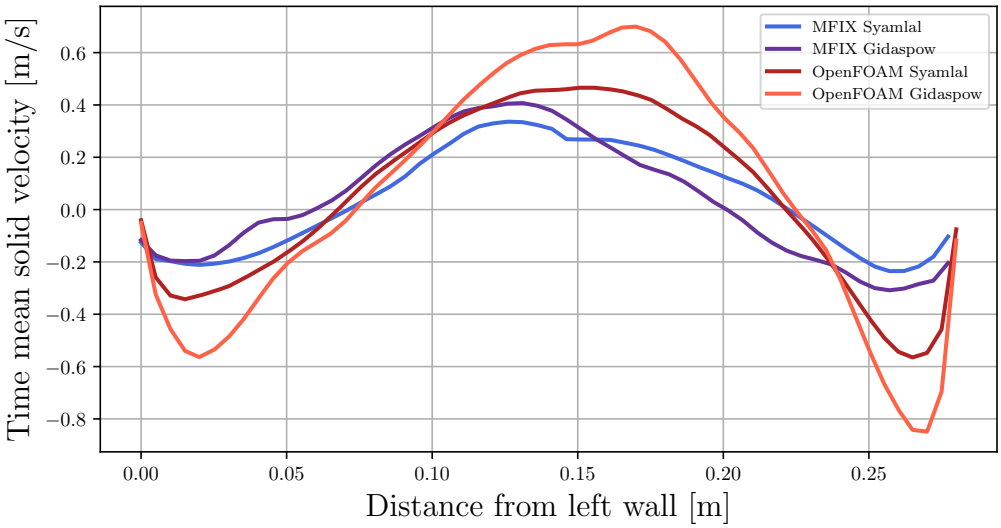


Figure 6.15: Time mean solid velocities measured in the range from 3 to 12 seconds with superficial gas velocity equal 0.38 m/s. The measurements are taken at y = 0.4 m.

Figure (6.16) shows the time averages voidage across the bed at 0.4 meters height with superficial gas velocity equal to 0.38 m/s for both drag models in MFiX and OpenFOAM. All lines can be seen to have the lowest voidage at the walls with increasing voidage towards the centre of the bed. There is some deviation however where the voidage increases drastically between the centre of the bed and the wall, this effect can be seen to be especially large in the results from OpenFOAM.

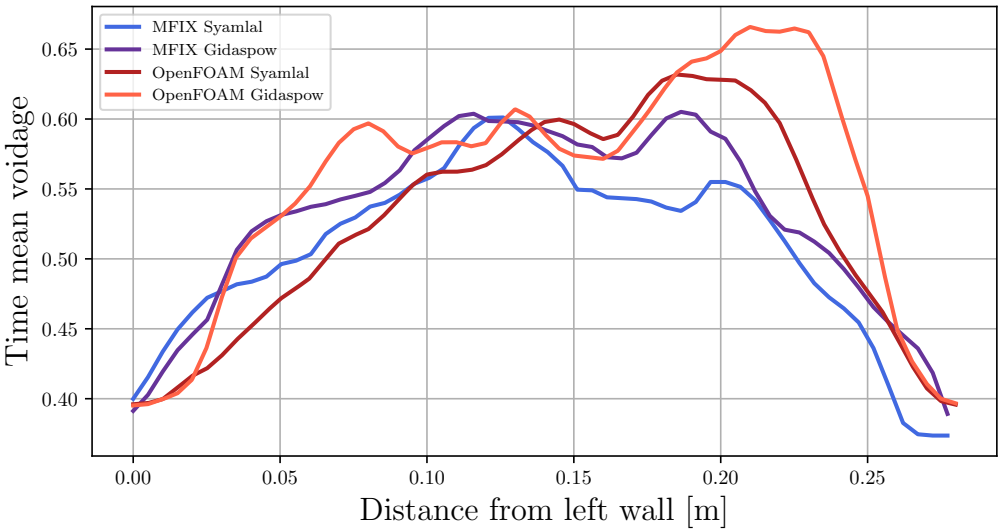
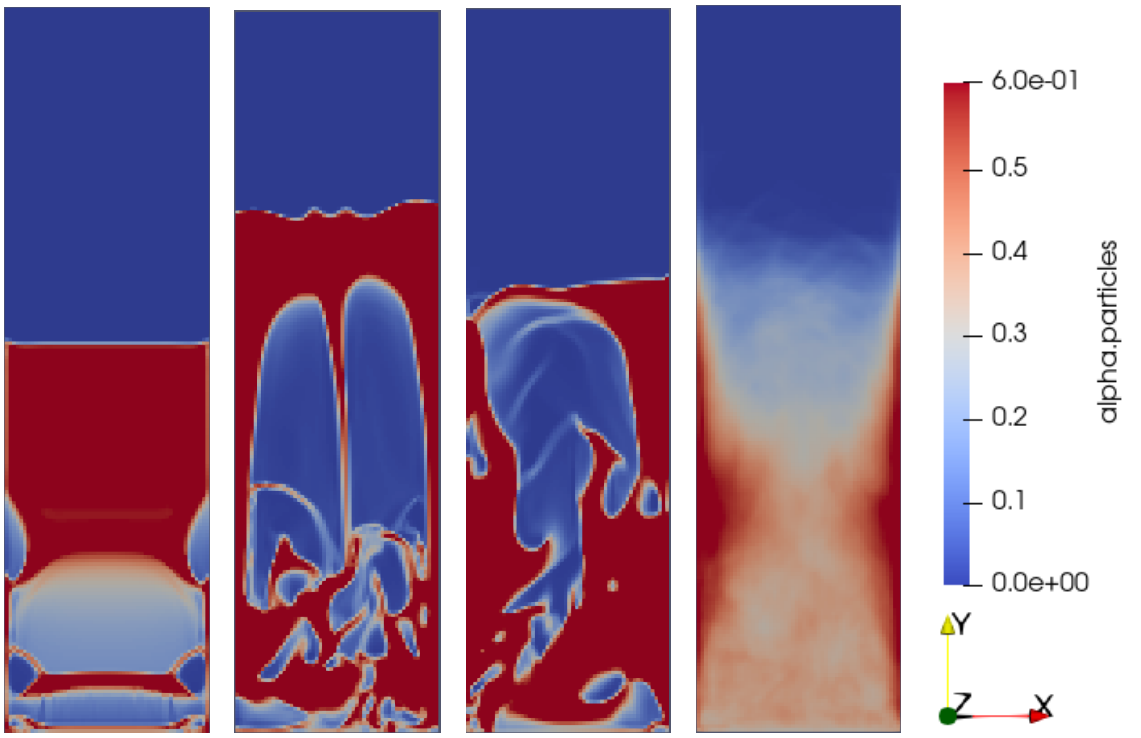


Figure 6.16: Time mean voidages measured in the range from 3 to 12 seconds with superficial gas velocity equal 0.38 m/s. The measurements are taken at y = 0.4 m.

6.5 Superficial gas velocity = 0.51 m/s

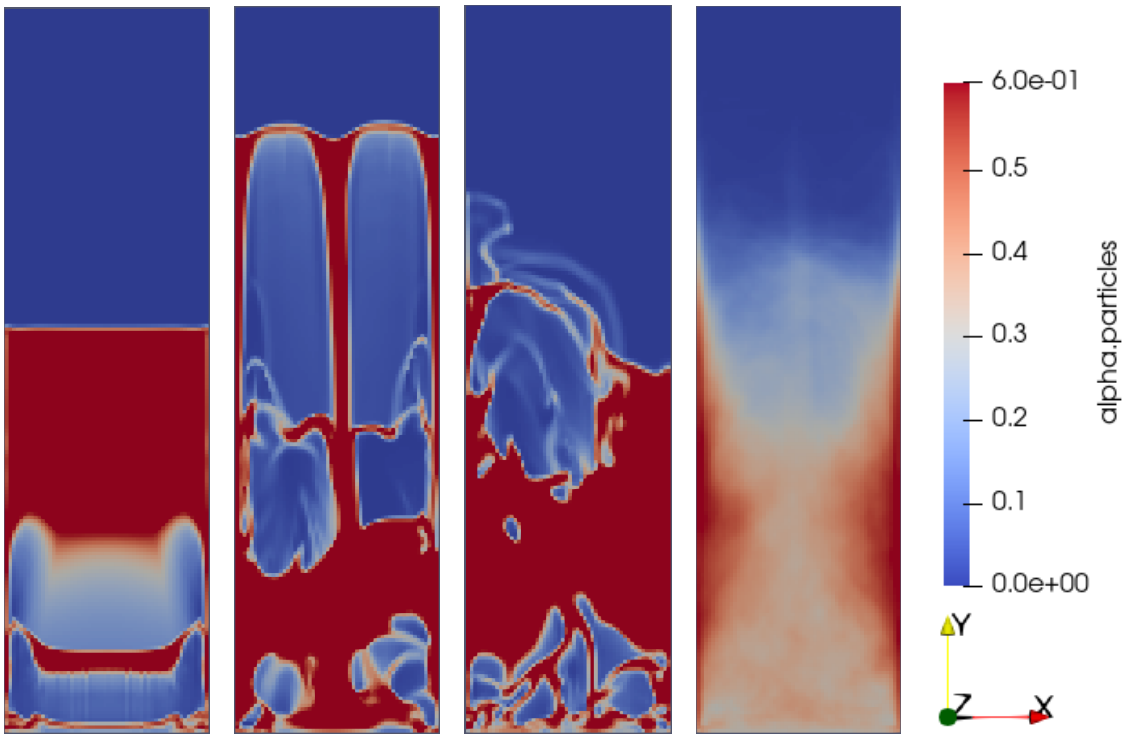
The evolution of the fluidized beds, as well as the steady state and time averaged solid volume fractions with a superficial gas velocity equal to 0.51 m/s can be seen in figures (6.17) and (6.18) for the Syamlal-O'Brien and Gidaspow drag models from MFiX, and figures (6.19) and 6.20 for the Syamlal-O'Brien and Gidaspow drag models from OpenFOAM. It can be seen from the figures that the bubbles that are created are larger and that the overall bed expansion is greater than those seen for the lower gas velocities.

The same differences between MFiX and OpenFOAM that occurred for the lower velocities can also be seen in these results, this being the distinct interface between the two faces in MFiX as seen in figures (6.17c) and (6.18c), which causes few and larger bubbles to be created, while the more diffuse interface observed in OpenFOAM in sub-figures (6.19c) and (6.20c) create smaller, but more abundant bubbles. In sub-figure (6.19d) it can be seen that OpenFOAM predicts a larger region of particle accumulation along the wall, like those from MFiX, seen in sub-figures (6.17d) and (6.18d) with the Syamlal-O'Brien drag model, while the Gidaspow model, seen in figure (6.20d) does not predict this, as was seen with OpenFOAM for the lower velocities. It can also be seen the Gidaspow model predicts a higher bed expansion than Syamlal-O'Brien in both softwares.



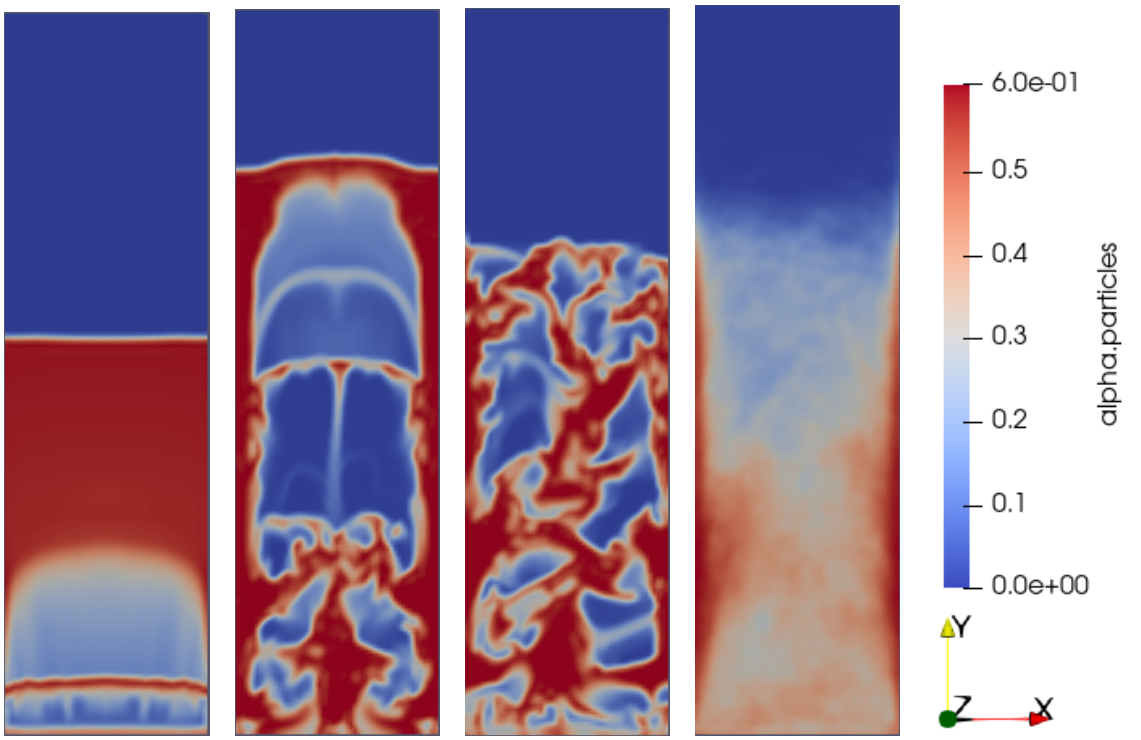
(a) 0.4 seconds (b) 1 second (c) Steady state (d) Time averaged (e) Scale bar

Figure 6.17: Particles distribution throughout the simulation with the Syamlal-O'Brien drag model with a superficial gas velocity equal to 0.51 m/s in MFiX.



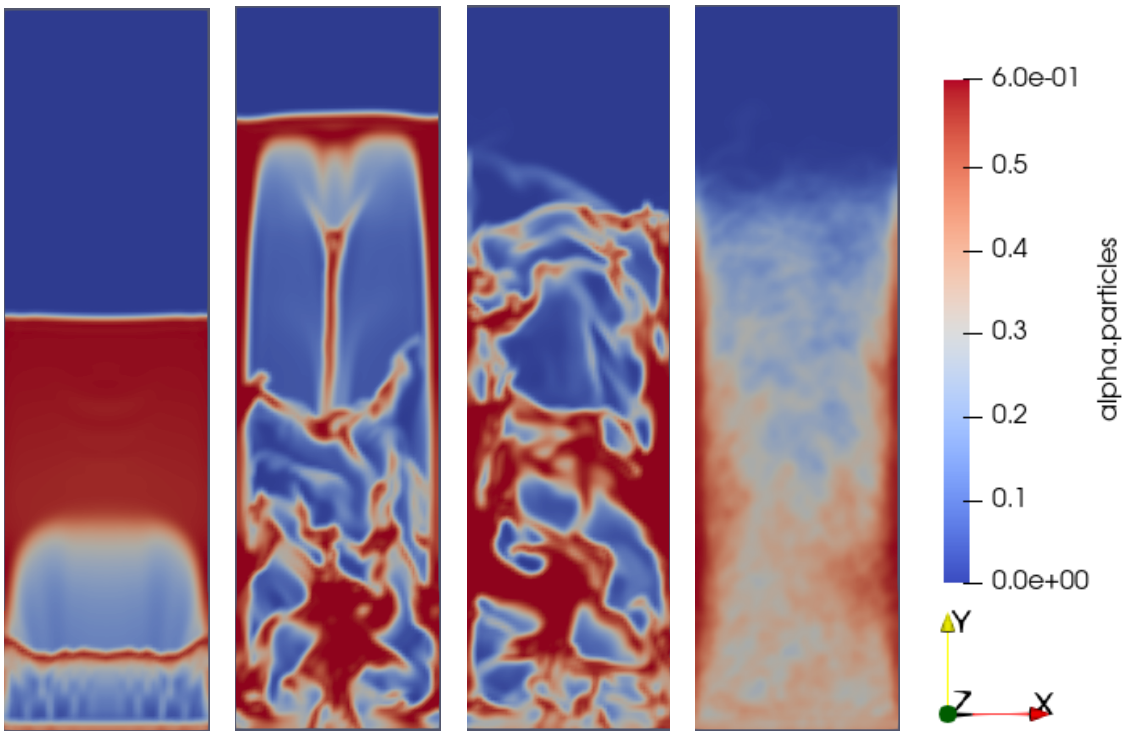
(a) 0.4 seconds (b) 1 second (c) Steady state (d) Time averaged (e) Scale bar

Figure 6.18: Particles distribution throughout the simulation with the Gidaspow drag model with a superficial gas velocity equal to 0.51 m/s in MFiX.



(a) 0.4 seconds (b) 1 second (c) Steady state (d) Time averaged (e) Scale bar

Figure 6.19: Particles distribution throughout the simulation with the Syamlal-O'Brien drag model with a superficial gas velocity equal to 0.51 m/s in OpenFOAM.



(a) 0.4 seconds (b) 1 second (c) Steady state (d) Time averaged (e) Scale bar

Figure 6.20: Particles distribution throughout the simulation with the Gidaspow drag model with a superficial gas velocity equal to 0.51 m/s in OpenFOAM.

The time average solid velocity in the y-direction at height of 0.4 meters with superficial gas velocity equal to 0.51 m/s can be seen in figure (6.21). The figure shows that all the results follow the same trend, with a parabolic form around the centre of the bed, and a sharp gradient at the wall. It can also be seen that OpenFOAM predicts a higher velocity which reaches above 0.75 m/s in both the downward and upwards direction, while MFiX predicts the largest velocity of roughly 0.5 m/s.

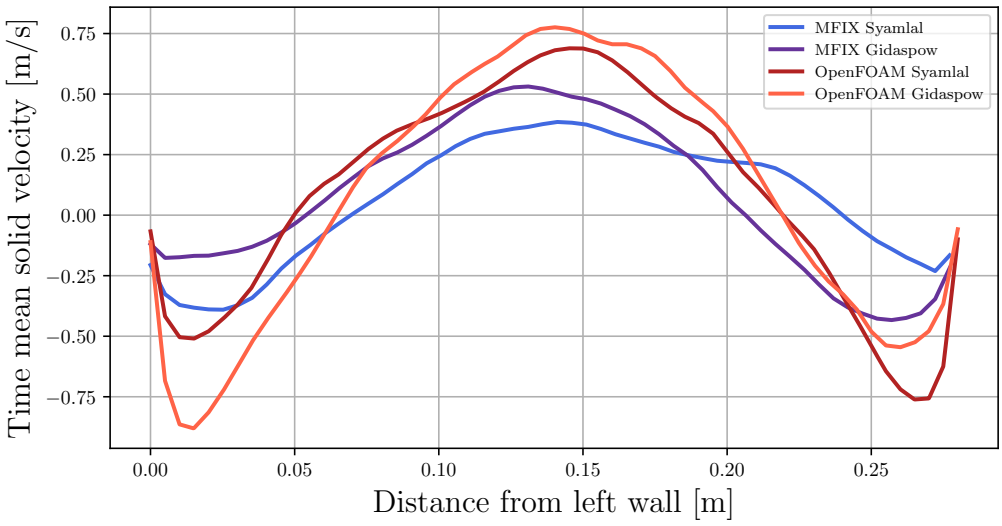


Figure 6.21: Time mean solid velocities measured in the range from 3 to 12 seconds with superficial gas velocity equal 0.51 m/s. The measurements are taken at y = 0.4 m.

Figure 6.22 shows the time average voidage at 0.4 meters height with superficial gas velocity equal

to 0.51 m/s. It can be seen from the figure that all the results follow the same pattern where the voidage is parabolic around the centre of the bed, with increasing voidage towards the middle of the bed. The results from OpenFOAM do however slightly deviate from this where the Gidaspow drag model has the largest voidage slightly left of the center and the Syamlal-O'Brien model has the largest voidage slightly right of the centre.

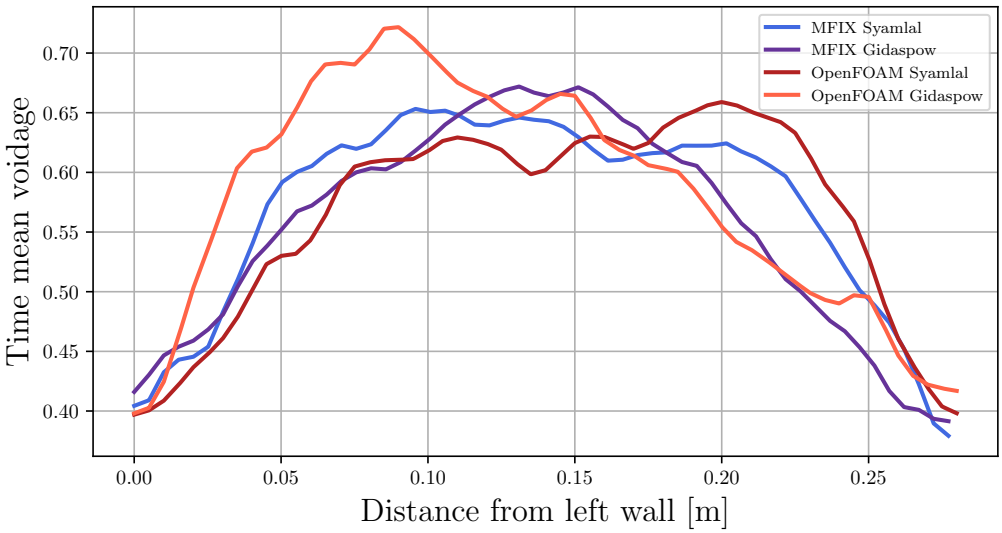
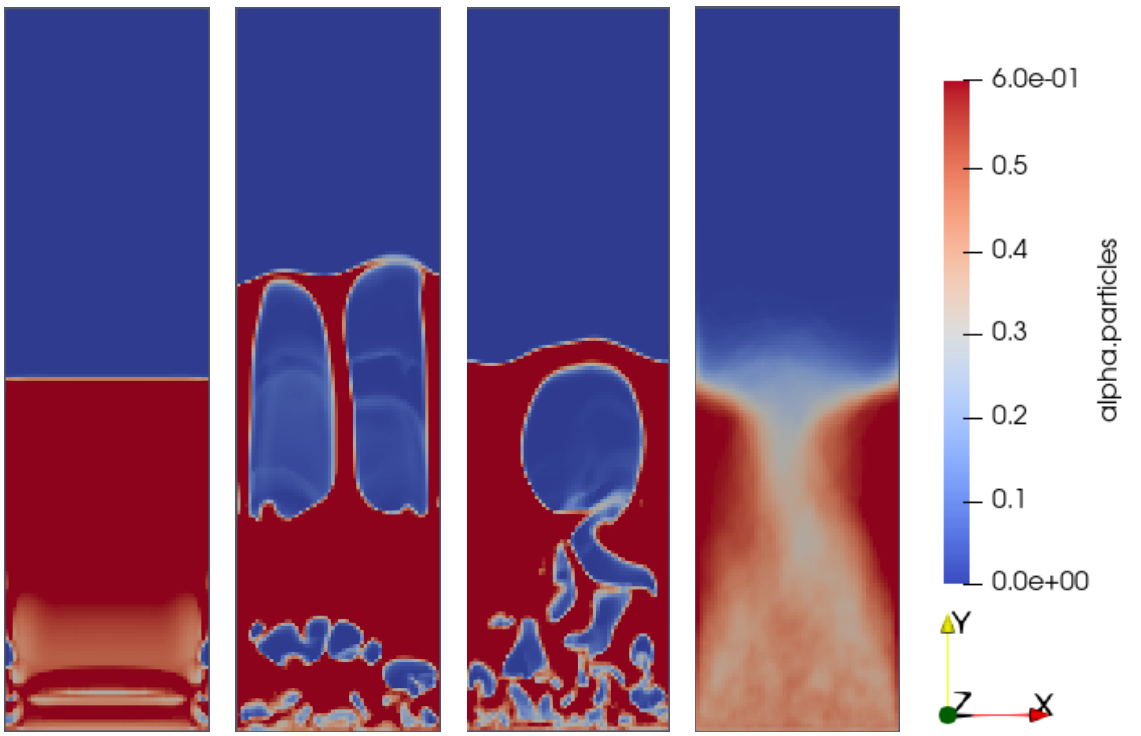


Figure 6.22: Time mean voidages measured in the range from 3 to 12 seconds with superficial gas velocity equal 0.51 m/s. The measurements are taken at y = 0.4 m.

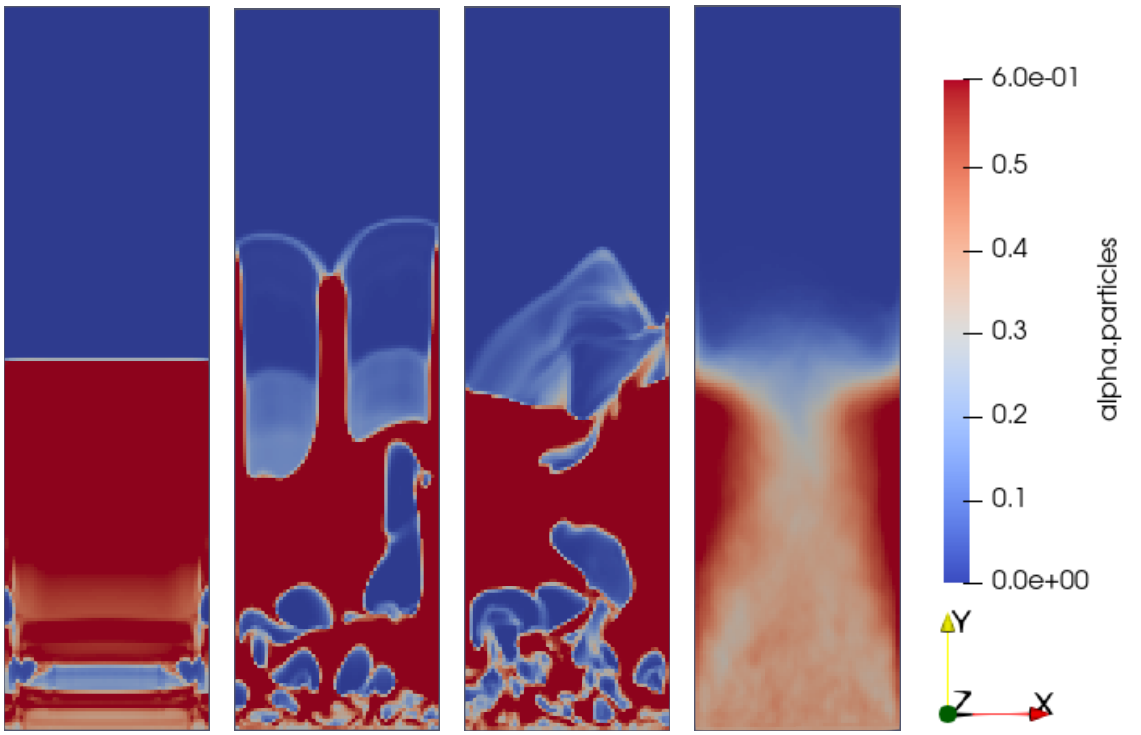
6.6 3D results

The evolution of the fluidized beds, as well as the steady state and time averaged solid volume fractions with a superficial gas velocity equal to 0.38 m/s for the middle of the z-axis in the 3D simulations can be seen in figures (6.23) and (6.24) for the Syamlal-O'Brien and Gidaspow drag models from MFiX, and figures (6.25) and (6.26) for the Syamlal-O'Brien and Gidaspow drag models from OpenFOAM. These results show the same trend as the previously discussed results, where there is a difference in the boundaries between the phases seen in MFiX and OpenFOAM with the former being distinct while the latter is more diffuse. The same pattern of MFiX having few but large bubbles, while OpenFOAM has smaller but more abundant bubbles can also be observed. A large accumulation of particles at the left and right walls can be seen in the results from MFiX while these regions remain comparatively small in the results from OpenFOAM.



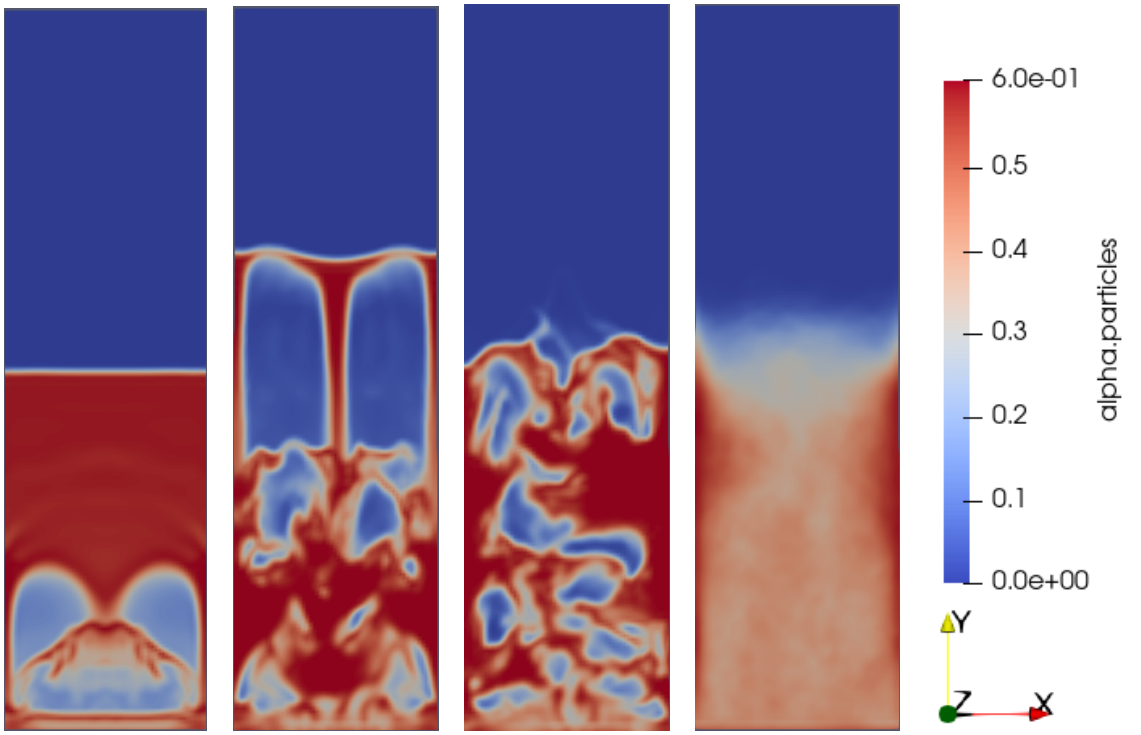
(a) 0.4 seconds (b) 1 second (c) Steady state (d) Time averaged (e) Scale bar

Figure 6.23: Particles distribution throughout the middle of the 3 dimensional simulation with the Syamlal-O'Brien drag model with a superficial gas velocity equal to 0.38 m/s in MFiX.



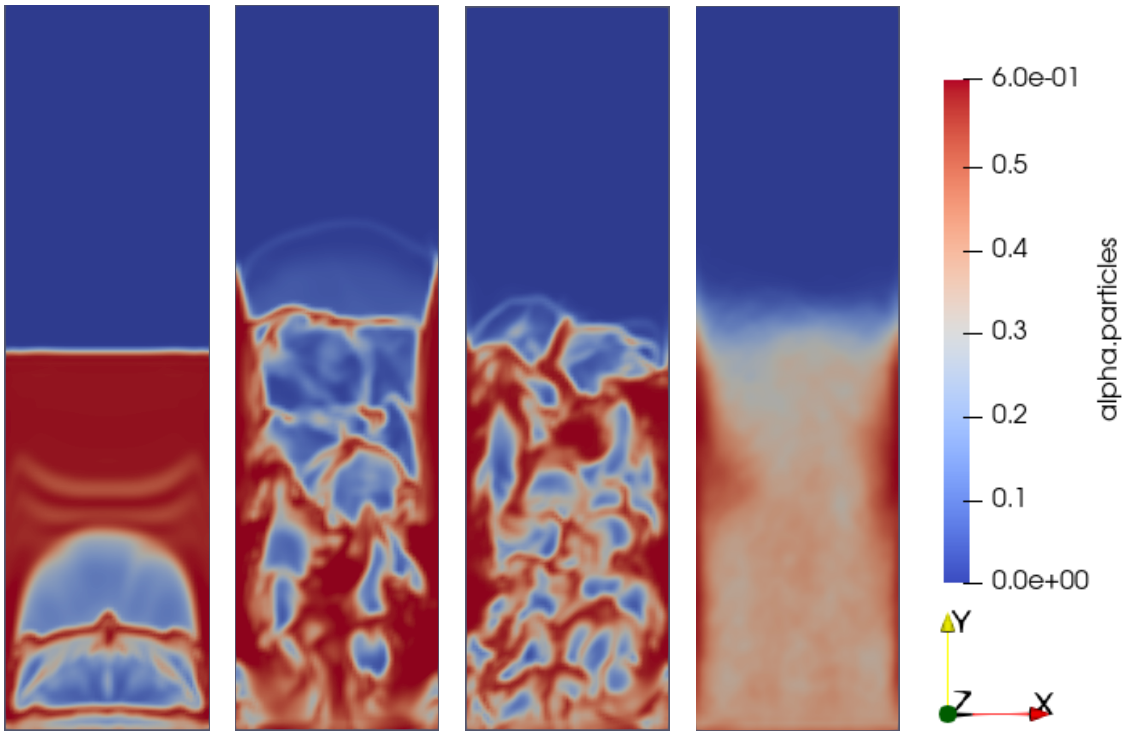
(a) 0.4 seconds (b) 1 second (c) Steady state (d) Time averaged (e) Scale bar

Figure 6.24: Particles distribution throughout the middle of the 3 dimensional simulation with the Gidaspow drag model with a superficial gas velocity equal to 0.38 m/s in MFiX.



(a) 0.4 seconds (b) 1 second (c) Steady state (d) Time averaged (e) Scale bar

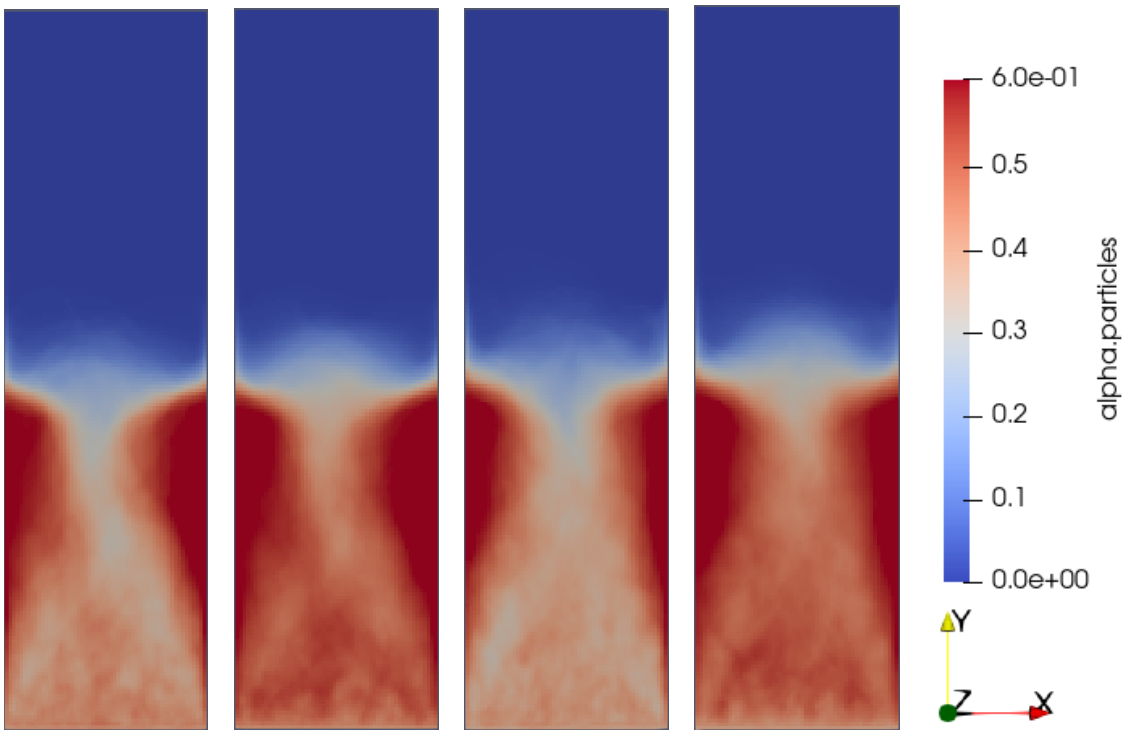
Figure 6.25: Particles distribution throughout the middle of the 3 dimensional simulation with the Syamlal-O'Brien drag model with a superficial gas velocity equal to 0.38 m/s in OpenFOAM.



(a) 0.4 seconds (b) 1 second (c) Steady state (d) Time averaged (e) Scale bar

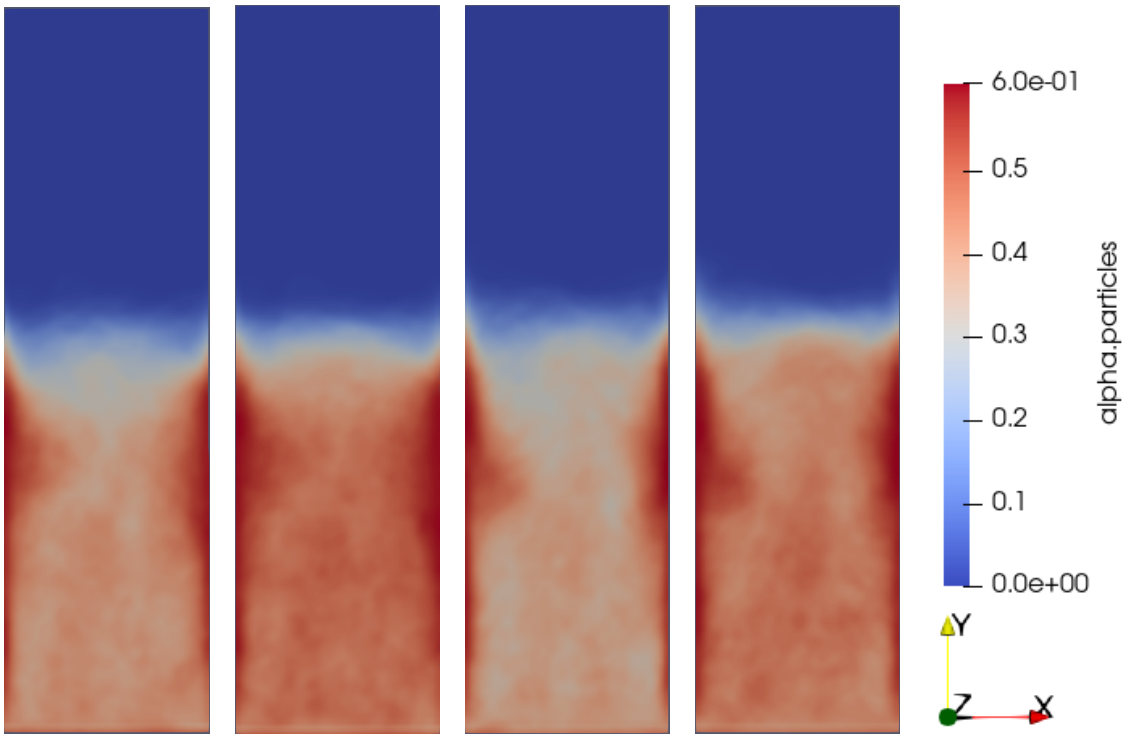
Figure 6.26: Particles distribution throughout the middle of the 3 dimensional simulation with the Gidaspow drag model with a superficial gas velocity equal to 0.38 m/s in OpenFOAM.

Figures (6.27) and (6.28) show the time averaged particle fractions in the middle of the bed and at the front wall for the Syamlal-O'Brien and Gidaspow drag models in MFiX and OpenFOAM respectively. It is clear from the figures that there is a larger amount of particles at the front wall rather than in the middle of bed for both models in both softwares. This effect does however appear to be stronger in the results obtained from MFiX, and that the gas can more easily pass at the wall in OpenFOAM.



(a) SO, middle. (b) SO, wall. (c) G, middle. (d) G, wall. (e) Scale bar

Figure 6.27: Time averaged particles distribution from the 3D simulations done in MFiX at the middle of the bed and at the front wall. Syamlal-O'Brien and Gidaspow are abbreviated as SO and G in the sub-figure captions.



(a) SO, middle. (b) SO, wall. (c) G, middle. (d) G, wall. (e) Scale bar

Figure 6.28: Time averaged particles distribution from the 3D simulations done in OpenFOAM at the middle of the bed and at the front wall. Syamlal-O'Brien and Gidaspow are abbreviated as SO and G in the sub-figure captions.

The time averaged solid velocity in the y-direction at 0.4 meters bed height with superficial gas velocity equal to 0.38 m/s can be seen for 2D and 3D in figure (6.29) for the Syamlal drag model and figure (6.30) for the Gidaspow drag model. The velocity in the 3D results were determined by taking the average of the velocities in each cell in the z-direction to get a result that is representative of the entire domain, like the 2D results. The figures show that the 3D results from MFiX mirror the 2D results. The 3D results from OpenFOAM on the other hand deviate significantly from the 2D results, where the 2D results have a parabolic shape, the 3D result grows to a certain value and stays relatively consistent throughout the entire bed.

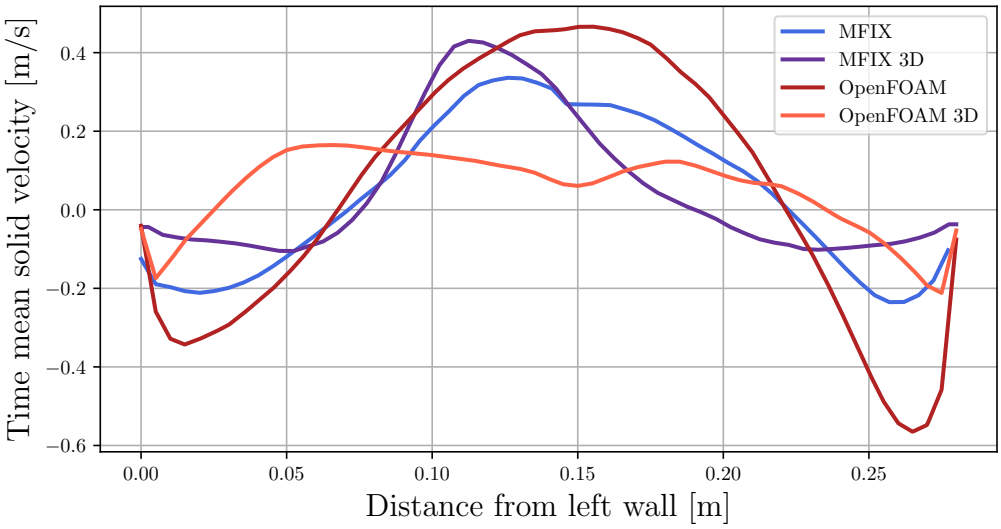


Figure 6.29: Comparison of the 2D and 3D time mean solid velocity measured with the Syamlal-O'Brien drag model with superficial gas velocity equal 0.38 m/s. Measurement is taken at y = 0.4 m.

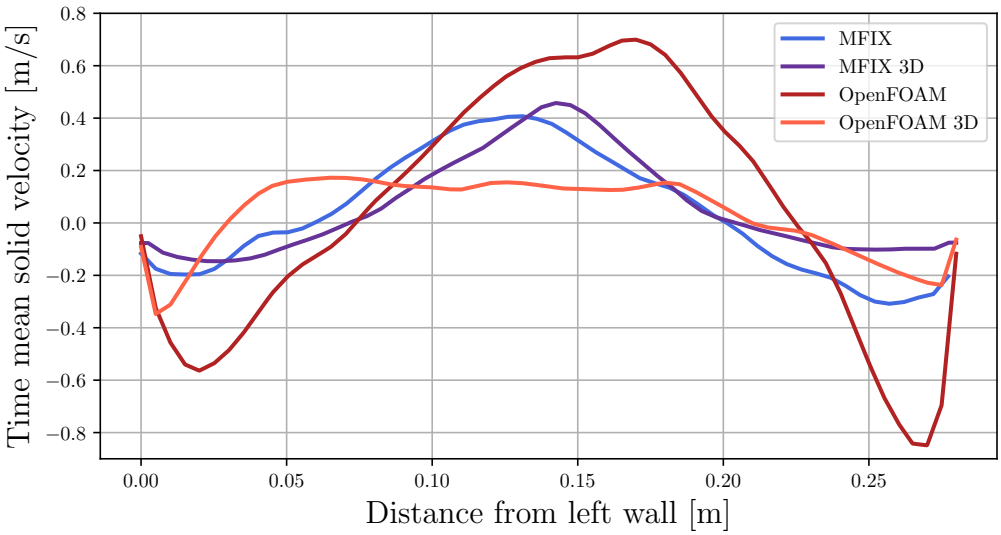


Figure 6.30: Comparison of the 2D and 3D time mean solid velocity measured with the Gidaspow drag model with superficial gas velocity equal 0.38 m/s. Measurement is taken at y = 0.4 m.

Figure (6.31) and (6.32) show the time averaged voidage results for the 2D and 3D simulations at 0.4 meter bed height with superficial gas velocity equal to 0.38 m/s with the Syamlal and Gidaspow drag models respectively. This figure shows that the effect of adding the 3rd dimension is opposite in OpenFOAM and MFiX. The 3D results from OpenFOAM has a lower maximum voidage than the 2D result across the bed but rapidly increases as it leaves the wall and stays relatively constant through the entire domain with both drag models. The opposite occurs for the 3D result in MFiX where the low voidage extends 5-10 cm out from the wall before rapidly increasing to a significantly higher peak than the 2D results in the middle of the bed, this happens with both the Syamlal-O'Brien and Gidaspow drag models.

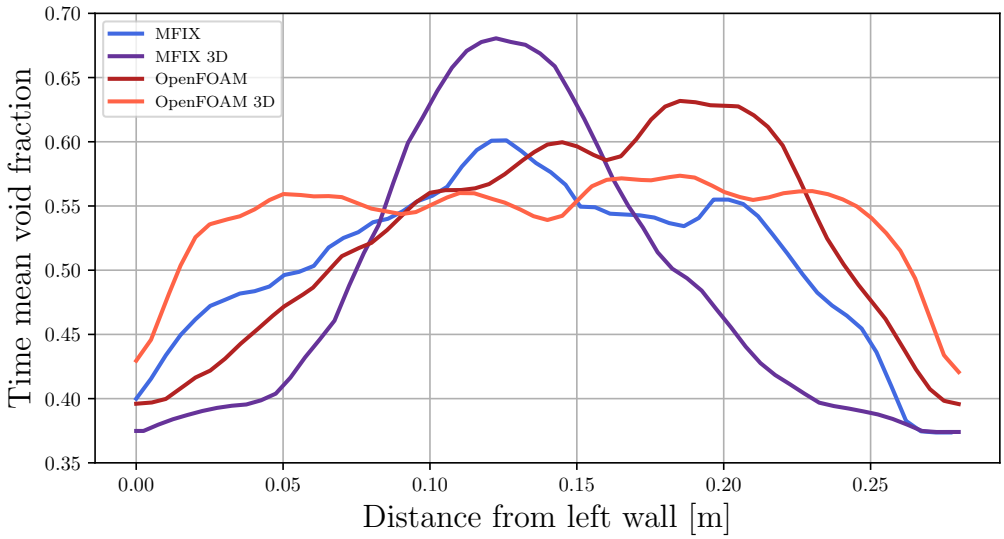


Figure 6.31: Comparison of the 2D and 3D time mean voidages measured with the Syamlal-O'Brien drag model with superficial gas velocity equal to 0.38 m/s. Measurement is taken at y = 0.4 m.

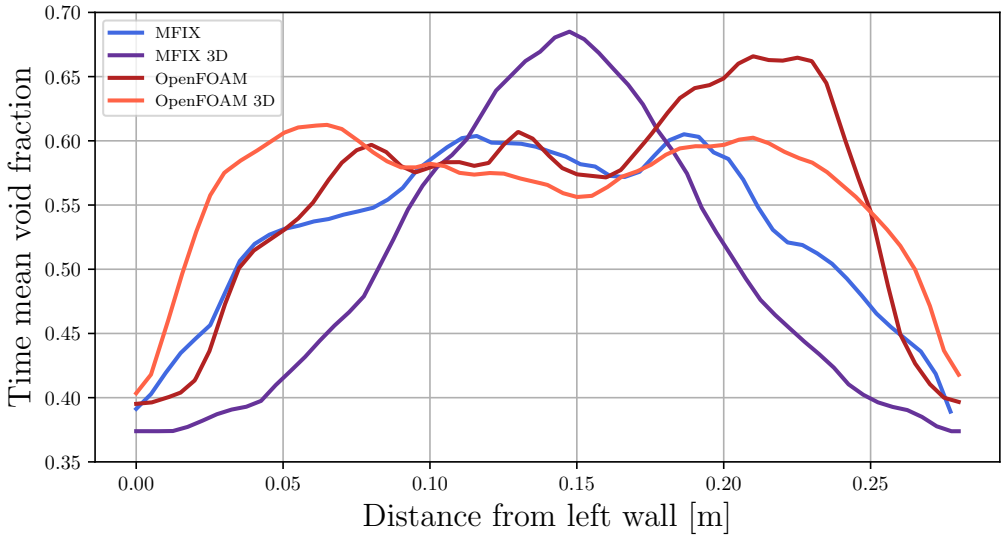


Figure 6.32: Comparison of the 2D and 3D time mean voidages measured with the Gidaspow drag model with superficial gas velocity equal to 0.38 m/s. Measurement is taken at y = 0.4 m.

7 Discussion

Cleaning of industrial gas emissions will continue to be an important process for years to come, it is therefore important to know that the tools utilized for developing and designing these processes are able to correctly approximate their behaviours. The present study has therefore been conducted with such a validation of the CFD codes in mind. MFiX has long been considered a "mature" predictor of fluidized bed, while OpenFOAM has been determined to be unreliable [18]. This perception appears to be pervasive throughout the field of computational fluid dynamics in regards to fluidized bed hydrodynamics as nearly all studies conducted either utilize FLUENT or MFiX, as can be seen from the literature review presented in section 2. The present study therefore conducts the same simulations in both MFiX and OpenFOAM to see if they predict the same hydrodynamic behaviours and are in general agreement with one another, and whether they are in agreement with the limited experimental data the author has at hand.

7.1 Comparison with experimental data

The findings from the present study will be compared with the experimental values found by Taghipour et al. [15], to see whether they are able to model the real world physical phenomena that occur. The pressure drops found and described earlier plotted together with the experimental data can be seen in figures (7.1) and (7.2) for the Syamlal-O'Brien and Gidaspow drag models respectively.

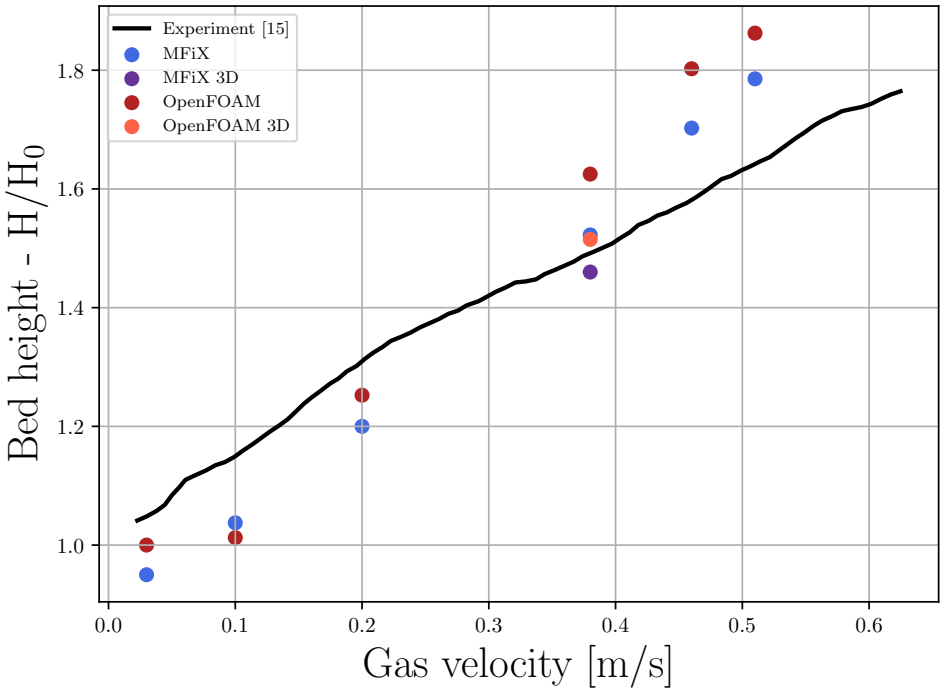


Figure 7.1: Bed expansion for different superficial gas velocities with the Syamlal-O'Brien drag model in both MFiX and OpenFOAM compared with the experimental data found by [15].

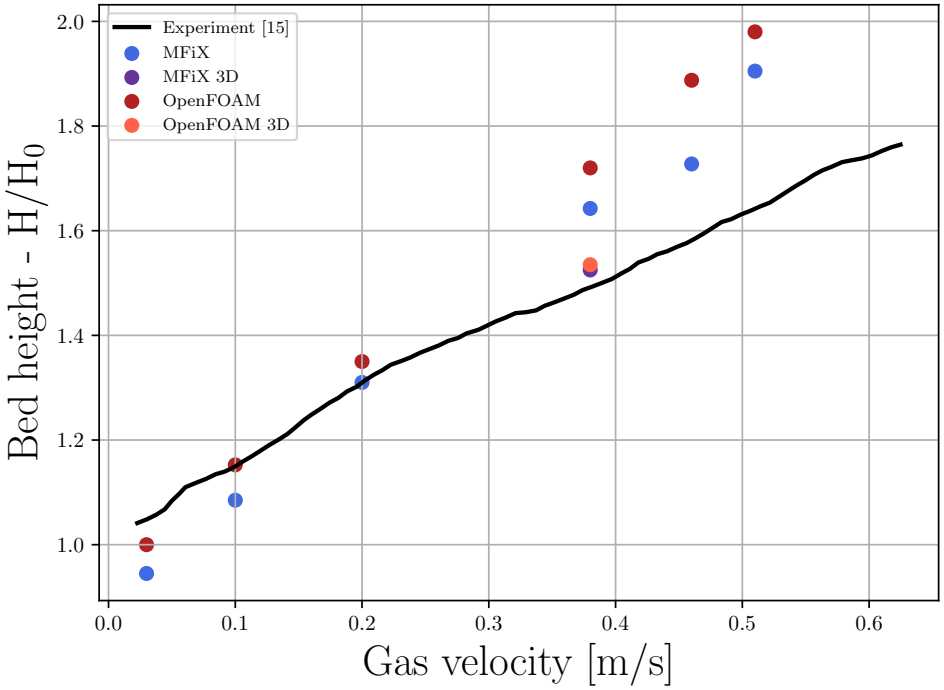


Figure 7.2: Bed expansion for different superficial gas velocities with the Gidaspow drag model in both MFiX and OpenFOAM compared with the experimental data found by [15].

The figures show that the Syamlal-O'Brien model underpredicts the bed expansion for superficial gas velocities lower than 0.2 m/s, while the Gidaspow drag model appears to be quite accurate for the lower velocities. Both drag models overpredict the bed expansion when the superficial gas velocity increases above 0.38 m/s, the Syamlal-O'Brien model is however in better agreement with the experimental data. All of the findings from the 3D simulations are in good agreement with the experimental data, which shows that the inclusion of friction along the front and back walls is important regardless of which drag model is being utilized. One issue with the experimental data is that it is not explicitly stated how the bed expansion was measured, so its validity in reference to the present findings can not be exactly determined, a similar trend between the simulated and experimental results can however be established.

The pressure drops obtained in the present study with the Syamlal-O'Brien and Gidaspow drag models are compared with the experimental pressure drop found by Taghipour et al. [15] and numerical findings by Herzog et al. [18] and can be found in figures (7.3) and (7.4) respectively. The experimental pressure drop can be seen to rapidly rise until it reaches the minimum fluidization velocity, after which it continues to increase slowly with increasing superficial gas velocity. The numerical findings are not in good agreement with this, they are seen to rapidly increase to the highest pressure drop around the minimum fluidization velocity and then decrease slowly with increasing superficial gas velocity.

When comparing the simulated results from the present study and those by Herzog et al [18], it can be seen that the results from MFiX are consistent between them, while the results from OpenFOAM are different. The previous findings from OpenFOAM show that it used to severely overpredict the pressure drop, while the present findings show that the results from OpenFOAM now follow the same trend as MFiX, but are around 500 Pa lower. The influence of adding the 3rd dimension can be seen to have opposite effects between MFiX and OpenFOAM with both drag models, where MFiX predicts a higher pressure drop and OpenFOAM a lower one, both of these appear to be more inaccurate than their 2D counterparts in relation to the experimental findings.

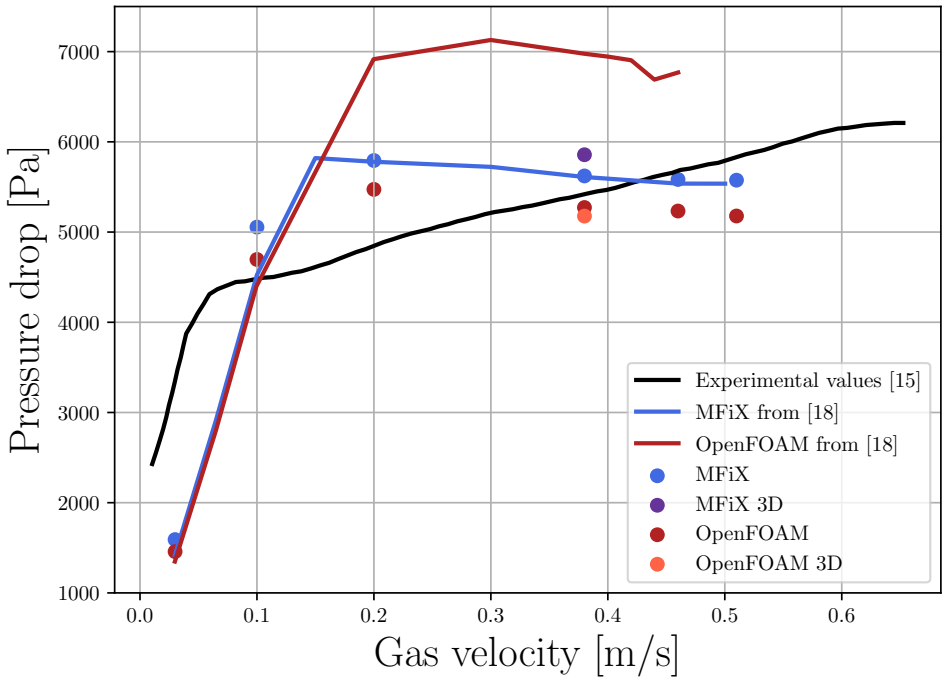


Figure 7.3: Pressure drop for different superficial gas velocities with the Syamlal-O'Brien drag model in both MFiX and OpenFOAM compared with the experimental values found by Taghipour et al. [15] and the simulated findings by Herzog et al. [18].

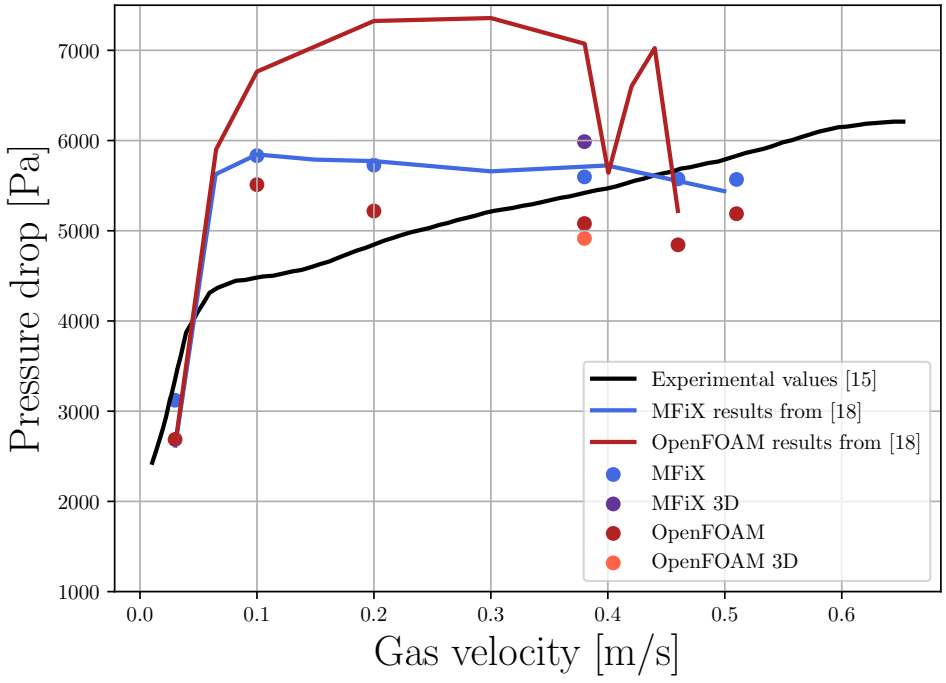


Figure 7.4: Pressure drop for different superficial gas velocities with the Gidaspow drag model in both MFiX and OpenFOAM compared with the experimental values found by Taghipour et al. [15] and the simulated findings by Herzog et al. [18].

Figures (7.5) and (7.6) show the time average void fractions for the 2D and 3D simulations at 0.38 m/s superficial gas velocity at 0.4 meters height together with the experimental data that was obtained by Taghipour et al. [15] for the Syamlal-O'Brien and Gidaspow drag models respectively.

It can be seen that none of the simulations give an exact match with the experimental data and that all the predicted void fractions are too large, the result could however be considered as having reasonable agreement with the experimental data. It can be seen that although the Syamlal-O'Brien and Gidaspow models give quite similar results that the Syamlal-O'Brien model is in slightly better agreement with the experimental data than Gidaspow.

Taghipour et al. explain in their paper that the experimental data show a preferred channel, which explains why the voidage is greater on the left side of the bed. The 2D results in OpenFOAM also appears to contain a preferred channel, this is however likely caused by the short simulation time and is expected to give a symmetric result if the simulation is allowed to run for longer and generate more data. The 3D results from MFiX with both drag models also show a preferred channel, but since this result is symmetric it is reasonable to assume that this would not change significantly with increased simulation time. The increased voidage that is seen for all the simulated results is in reasonable agreement with the previously discussed result as one would expect a larger void fraction in a fluidized reactor with higher bed expansion.

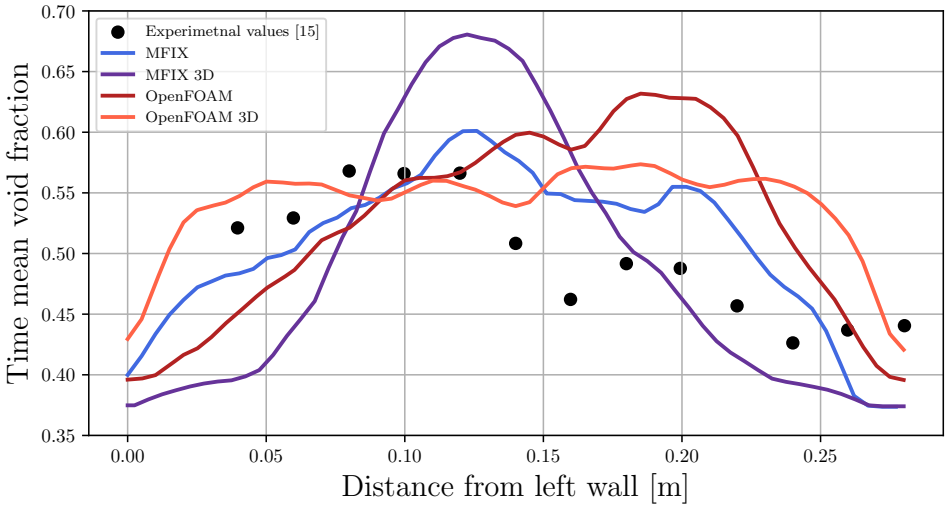


Figure 7.5: Comparison of the experimental voidage found by Taghipour et al. [15] and the 2D and 3D time mean voidages measured with the Syamlal-O'Brien drag model with superficial gas velocity equal to 0.38 m/s. Measurement is taken at y = 0.4 m.

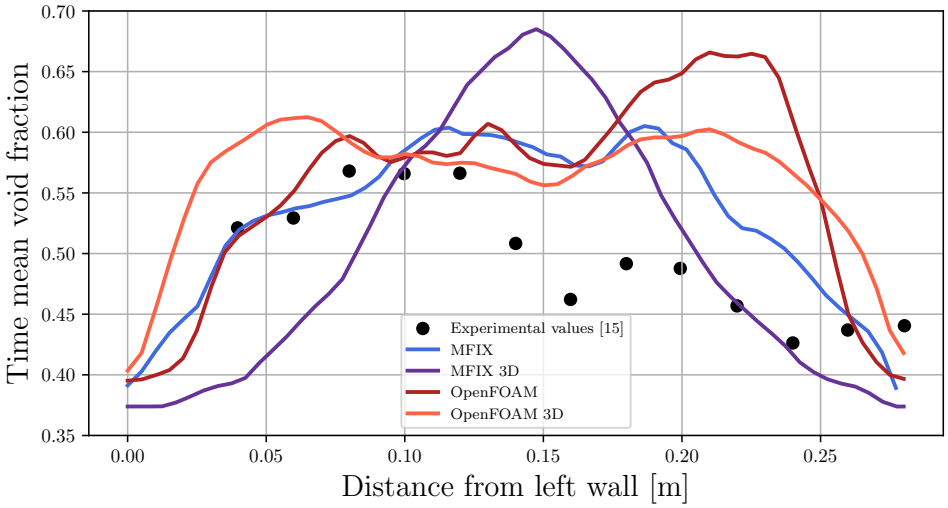


Figure 7.6: Comparison of the experimental voidage found by Taghipour et al. [15] and the 2D and 3D time mean voidages measured with the Gidaspow drag model with superficial gas velocity equal to 0.38 m/s. Measurement is taken at y = 0.4 m.

7.2 Model limitations

The pressure drop in a fluidized bed as described by the literature is said to rapidly increase to a maximum value when it reaches the minimum fluidization velocity, before staying relatively constant, or slightly decreasing/varying due to slugging behaviours as the superficial gas velocity increases [8]. This behaviour is what the drag models are trying to replicate and can clearly be seen in the results in figures (6.1) and (6.2). This behaviour is not what is encountered in the experimental data from Taghipour et al. [15], as was previously discussed. In the experiment the pressure drop continuously increases with increasing superficial gas velocity, this means that the drag models will not be able to perfectly replicate the experimental data as the models are incongruent with the actual physical phenomenon.

The Eulerian-Eulerian model considers both the solid and gas phases as continua, this is a compromise that is made due to computational convenience that loses some of the particle characteristics due to the implementation of the kinetic theory of granular flow. The kinetic theory of granular flow, which governs the constitutive relations of the solid phase is based on several approximations and assumptions, that are not entirely realistic. The first and one of the biggest assumptions is that all particles are spherical, exactly the same shape and have the same orientation. The model also has several empirical relations to approximate phenomena such as for example particle collisions and particle-boundary interactions. Many of the equations and relations utilized in the kinetic theory of granular flow contain variables from kinetic theory, such as granular temperature, solid pressure or solid shear viscosity, all of which are quantities that can not be measured in the real world, which then limits the possibility of validation of the model.

The Johnson-Jackson boundary condition which determines how the granular phase interacts with the wall boundaries depends on two main variables, the specularity and restitution coefficient. A limitation of this boundary condition is that both of these coefficients have great influence on not only the interaction with the wall, but the hydrodynamics of the entire bed, and are very difficult to determine and can vary with the superficial gas velocity [22]. This limitation is reflected in the literature study conducted in this thesis where 7 out of the 10 studies that were looked at investigated the effect of either the specularity or restitution coefficient and tried to determine an appropriate value for them.

There is also a limit to how refined the mesh can be when utilizing the Eulerian-Eulerian model. This is because the models from the kinetic theory of granular flow are not able to properly predict the behaviour of the particle fraction when there are too few particles present in a given control volume. The result of having a grid size that is too refined, typically smaller than 10 times the particle diameter is large perturbations and fluctuations in the solid particle phase. This phenomenon was looked into by Kong et al. [21] and was found to cause unrealistic results.

There are also some shortcomings with both the Gidaspow and Syamlal-O'Brien drag models as have been demonstrated in this thesis. It can be seen that while the Gidaspow model gives good predictions for the lower superficial gas velocities tested i.e. 0.03, 0.1 and 0.2, it does however overpredict the hydrodynamics for velocities larger than this. The Syamlal-O'Brien drag model is the opposite of this where it severely underpredicts for the lowest superficial gas velocities, but gives much better predictions than Gidaspow for the larger ones. This is in good agreement with the study by Bakshi et al. [23] who reported similar findings with the two drag models in their work.

The reason why the models predict different results is likely due to the ways they are derived. The Gidaspow model, being based on the Ergun equation and Wen-Yu model, which relates the pressure drop in the bed to the drag coefficient, fails to account for turbulent forces and increased drag around the particles at increased velocities, leading to an overprediction. The Syamlal-O'Brien model on the other hand utilizes a relation between the terminal velocity of a particle to derive a relation for the drag coefficient. The way the Syamlal-O'Brien model is configured by default leads to a minimum fluidization velocity which will not be correct for all particles, it is for this reason the Syamlal-O'Brien model needs a larger velocity before it begins predicting properly as was seen from the results in this thesis.

7.3 Numerical settings

The numerical settings used in the simulations are the ones that are presented as the defaults in the preset cases in the respective softwares. These are used because they are chosen by the developers and are therefore assumed by the author to give reasonable results, and a good assessment on the general capabilities of the software, without the need to do significant tinkering before starting simulations. This does, however, mean that the numerical settings between the softwares are different, as can be seen when comparing tables 5.4 and 5.5 with figures (5.4) and (5.5) which show the numerical settings used in OpenFOAM and MFiX respectively.

There are several differences between the numerical settings in the two softwares, first of all, the discretization schemes utilized. MFiX uses a high order upwind scheme with a downwind factor with the Superbee flux limiter, additionally, it also applies a relaxation factor of 0.8 or 0.5 to all relevant variables that are being solved. OpenFOAM on the other hand uses Gaussian discretization, and no relaxation factor, with different flux limiters like "vanLeer" or "limitedLinear" depending on the variable. The linear solver that is used in MFiX is BiCGStab which is used for all variables, the tolerance that is chosen by default is quite high and is set to 10^{-4}. OpenFOAM uses a range of linear solvers with varying complexities and has a much stricter tolerance than MFiX at either 10^{-8} or 10^{-9} for the relevant variables.

The differences between the numerical settings necessarily mean that the results that are obtained from the two softwares will be different. It is however difficult to say which settings have the biggest impact. The fact that the solution algorithms, which were explained in section 4.1.6, are also quite different complicates this matter further.

The numerical schemes that are used in MFiX are of higher order than those in used in OpenFOAM in the current study. In addition to this, the flux limiter in MFiX, Superbee, is known for its high stability, especially in regions where there are large gradients, like when a bubble collapses, and discontinuous regions like the phase interfaces. The schemes in OpenFOAM are of 2nd order in addition to using the limitedLinear flux limiter for most variables, and van Leer for the solid particle fraction. The lower order schemes used in OpenFOAM could therefore introduce numerical diffusivity to the solution, even though the tolerances are tighter than MFiX. This could explain why the diffuse borders between the phases could be observed in OpenFOAM, but not in MFiX.

Another aspect that likely also affects the difference in interfaces between the phases in the two softwares could be the solution algorithms that are utilized, as they not only compute different variables in different orders but also use different calculations. The OpenFOAM solver uses the MULES sub-cycle to handle the different phases, this algorithm is originally developed for the volume of fluid (VOF) method, which could lead to undesired behaviours when it is applied to an Eulerian-Eulerian model as it was not designed for this. It is therefore difficult to pinpoint whether the solution algorithm or the numerical settings or a combination of the two is the reason for the differing simulated behaviours.

7.4 OpenFOAM vs MFiX

The previous section looked at the differences between the numerical settings in the two softwares and how these could potentially affect the results and to how large of a degree. The coming discussion will be focused on exactly how the two different softwares have predicted the fluidized reaction and how these differences propagate through the entire result.

7.4.1 Overall evaluation

The differences that can be seen in the flow evolutions and behaviours between MFiX and OpenFOAM can likely explain all of the differences that are seen between the results in the two softwares. The biggest difference in the flow evolutions, which has been mentioned and discussed previously is the differences in phase interfaces between the two softwares. The plentiful and diffuse bubbles that are formed in OpenFOAM, means that there is gas flowing and pushing the particles upwards

across almost the entire domain. This can be seen when looking at the time averaged plots from OpenFOAM, where there is only a thin region of particle accumulation along the wall, where the particles are able to move downward.

The same results from MFiX show a different trend, the small bubbles that are formed in the bottom of the bed combine together into relatively larger and fewer distinct bubbles that rise through the centre of the bed. This means that there is channelling in the middle of the bed in MFiX, which can not be observed to the same degree in OpenFOAM. This channeling does however allow for a larger region of particles to move downward in the bed, which can be seen in the time averaged plots from MFiX, which shows an accumulation of particles that stretches some distance away from the all.

These differences between OpenFOAM and MFiX could help explain three different phenomena that were observed in the result section. The first phenomenon being that there is greater bed expansion in OpenFOAM than in MFiX, as the small bubbles allow for less downward motion, and force the solid particles upwards at a larger velocity. The second phenomenon being that the pressure drop measured in OpenFOAM being lower than MFiX, this is because the slugging behaviour that is caused by the more numerous collapsing bubbles causes the pressure drop to decrease, which is in agreement with the theory. The third phenomenon being related to the first one, which is the relatively smaller region the particles can move in, in OpenFOAM. This forces the particles to move rapidly through this region if continuity is to be upheld, which explains why the downward velocity measured in OpenFOAM is greater and has a larger gradient than for the same simulations in MFiX.

7.4.2 Effect of 3D

The inclusion of the front and back wall in the simulation has a large impact on the simulated result. The differences between OpenFOAM and MFiX that were observed in the 2D simulations stay the same but they are increased to an even greater degree. The hint of a preferred channel that was observed in MFiX in 2D becomes extremely pronounced and occurs exactly in the middle of the domain with the inclusion of the front and back walls. The ability for particles to accumulate on the front and back walls also allows for particles on the left and right walls to extend even further into the bed. This leads to a slow downward velocity of the particles that extends far out from the walls and upwards velocity only in the absolute centre. The preferred channel can also be seen in the voidage plots where a voidage of 0.4 i.e. initial packing extends more than 5 cm out from the wall. The smaller diameter of the preferred channel is likely the cause of the increased pressure drop observed in MFiX.

The exact opposite happens in OpenFOAM where the thin layer along the left and right walls stays small, and a thin layer of particles accumulates on the front and back walls. The addition of the thin layer of particles on the front and left wall means that particles can now move downward along the entire x-axis. The effect this appears to have is that solid velocity and voidage is that they can stay relatively constant across the entire bed without the same need of pushing particles to the left or right walls. The pressure drop stays approximately equal between 2D and 3D in OpenFOAM and the difference could be caused due to a lack of simulation time.

7.4.3 Johnson-Jackson boundary condition

The Johnson-Jackson wall boundary condition has been applied in the same way with the same coefficients of specularity and restitution in both softwares and should therefore give the same results. Loha et al. [19, 20] found that different values of the coefficients lead to differing amounts and sizes of bubbles in addition to changing the velocities that could be observed along the boundary walls. Both of these phenomena are the biggest differences between OpenFOAM and MFiX. One can therefore not rule out that potentially differing implementations of this boundary condition in the source code is one of the reasons for the variations between the results.

8 Conclusion

In this thesis the background theory for fluidized beds has been presented in addition to the background theory of how computational fluid dynamics works and how the fluidized bed models are implemented. The present study has looked at the ability of MFiX, which is a validated and trusted implementation specialized in fluidization, and OpenFOAM have been compared with existing experimental data and each other. Certain limitations of the Eulerian-Eulerian framework and fluidized bed models have also been discussed. The present findings show that:

- MFiX and OpenFOAM are both able to compute results that are in reasonable agreement with the experimental data with both drag models that give a general overview of the hydrodynamics in the fluidized beds.
- The Gidaspow and Syamlal-O'Brien drag models both give good predictions but for different conditions. The Gidaspow drag model is more suited for velocities that are close to the minimum fluidization velocity (U_{mf}), while the Syamlal-O'Brien model gives better agreement for superficial gas velocities above $5 \cdot U_{mf}$.
- The Eulerian-Eulerian model has certain limitations due to underlying assumptions and approximation. This hinders the results from being in exact agreement with experimental findings, but it is a computationally effective way of getting a general overview of the fluidized bed hydrodynamics.
- OpenFOAM and MFiX handle the interfaces between the phases present in the simulations in a different manner, which leads to the two softwares producing different results. It is however difficult to determine whether the numerical settings, solution algorithms, implementation of boundary conditions or a combination of the three is the reason for this, and further studies are required to pinpoint the exact sources.
- The inclusion of the front and back walls by doing 3D simulations are important as the friction along these walls significantly alters the results. This does however increase the computational time significantly which means 3D studies should be targeted on a very specific topic.
- Overall there is a marked improvement in the results obtained from OpenFOAM compared to previous studies. OpenFOAM appears as being mature to predict fluidized bed phenomena although further efforts are needed to validate and understand how different numerical settings and boundary conditions affect the results it produces.

9 Future work

The experimental data on the fluidization phenomena in relation to aluminium production is currently limited, and the ability of CFD software to predict the hydrodynamics of small particles like alumina is unknown. The following future work related to this is therefore proposed:

- Conduct an experimental study which explores the hydrodynamics of alumina particles in conditions that are similar to ones found in industrial processes.
- Conduct an experimental study which looks at the influence of superficial gas velocity on bubble size and quantity and try to develop corrections to the drag laws and/or boundary conditions.
- In addition, one should; conduct an experimental study that incorporates the chemical reactions between alumina and industrial off-gas to see how this influences the hydrodynamics.

The validation of OpenFOAM as a predictor of fluidization phenomena is in its beginning stages and further studies are needed to understand its inner workings. Proposed research questions in relation to this are:

- Conduct a similar study to this one with OpenFOAM, where only the numerical settings are altered to find out how they affect the final result.
- Conduct a study which implements the potential empirical relation discussed previously.
- Study the influence of the restitution and specularity coefficients in OpenFOAM and relate them to similar findings in FLUENT and MFiX.

Bibliography

[1] CRU International. *Opportunities For Aluminium In A Post-Covid Economy*. Tech. rep. International Aluminium Institute, 2022.

[2] A. Solheim. *Aluminium Electrolysis Main Principles and Technology*. Tech. rep. SINTEF Materials and Chemistry, 2009.

[3] T. A. Aarhaug and A. P. Ratvik. 'Aluminium Primary Production Off-Gas Composition and Emissions: An Overview'. In: *JOM* 71.9 (Sept. 2019), pp. 2966–2977. ISSN: 1543-1851. DOI: 10.1007/s11837-019-03370-6. URL: https://doi.org/10.1007/s11837-019-03370-6.

[4] O. Lorentsen and M. Tangstad. *Metal production in Norway - Aluminium*. First. Trondheim: Akademika forlag, 2013.

[5] P. M. Witt and D. A. Hickman. 'Fluidized-bed reactor scale-up: Reaction kinetics required'. In: *AIChE Journal* 68.9 (2022), e17803. DOI: https://doi.org/10.1002/aic.17803. eprint: https://aiche.onlinelibrary.wiley.com/doi/pdf/10.1002/aic.17803. URL: https://aiche.onlinelibrary.wiley.com/doi/abs/10.1002/aic.17803.

[6] J. W. Slater. *Validation Assessment*. 2021. URL: https://www.grc.nasa.gov/www/wind/valid/tutorial/valassess.html (visited on 19/04/2023).

[7] R. M. Felder and R. W. Rousseau. *Elementary Principles of Chemical Processes*. Third. USA: Wiley and Sons, 2000.

[8] J. Szekely, J. W. Evans and H. Y. Sohn. *Gas-Solid Reactions*. First. 111 Fifth Avenue, NY: Academic Press, 1976.

[9] S. Ergun. 'Fluid flow through packed columns'. In: *Chem. Eng. Prog.* 48.2 (1952), pp. 89–94.

[10] R. Newell and N. Standish. 'Velocity Distribution in Rectangular Packed Beds and Non-Ferrous Blast Furnaces'. In: *Metallurgical Transactions* 4 (1973), pp. 1851–1857.

[11] J.F. Davidson and D. Harrison. *Fluidised Particles*. Cambridge: Cambridge University Press, 1963.

[12] S. Karimipour and T. Pugsley. 'A critical evaluation of literature correlations for predicting bubble size and velocity in gas–solid fluidized beds'. In: *Powder Technology* 205.1 (2011), pp. 1–14. ISSN: 0032-5910. DOI: https://doi.org/10.1016/j.powtec.2010.09.016. URL: https://www.sciencedirect.com/science/article/pii/S0032591010004936.

[13] L. Huilin et al. 'Size segregation of binary mixture of solids in bubbling fluidized beds'. In: *Powder Technology* 134.1 (2003), pp. 86–97. ISSN: 0032-5910. DOI: https://doi.org/10.1016/S0032-5910(03)00126-8. URL: https://www.sciencedirect.com/science/article/pii/S0032591003001268.

[14] W. Zhong et al. 'CFD simulation of dense particulate reaction system: Approaches, recent advances and applications'. In: *Chemical Engineering Science* 140 (2016), pp. 16–43. ISSN: 0009-2509. DOI: https://doi.org/10.1016/j.ces.2015.09.035. URL: https://www.sciencedirect.com/science/article/pii/S0009250915006661.

[15] F. Taghipour, N. Ellis and C. Wong. 'Experimental and computational study of gas–solid fluidized bed hydrodynamics'. In: *Chemical Engineering Science* 60.24 (2005), pp. 6857–6867. ISSN: 0009-2509. DOI: https://doi.org/10.1016/j.ces.2005.05.044. URL: https://www.sciencedirect.com/science/article/pii/S0009250905004653.

[16] J. T. Cornelissen et al. 'CFD modelling of a liquid–solid fluidized bed'. In: *Chemical Engineering Science* 62.22 (2007), pp. 6334–6348. ISSN: 0009-2509. DOI: https://doi.org/10.1016/j.ces.2007.07.014. URL: https://www.sciencedirect.com/science/article/pii/S0009250907005477.

[17] T. Li, J. Grace and X. Bi. 'Study of wall boundary condition in numerical simulations of bubbling fluidized beds'. In: *Powder Technology* 203.3 (2010), pp. 447–457. ISSN: 0032-5910. DOI: https://doi.org/10.1016/j.powtec.2010.06.005. URL: https://www.sciencedirect.com/science/article/pii/S0032591010003049.

[18] N. Herzog et al. 'A comparative study of different CFD-codes for numerical simulation of gas–solid fluidized bed hydrodynamics'. In: *Computers & Chemical Engineering* 39 (2012), pp. 41–46. ISSN: 0098-1354. DOI: https://doi.org/10.1016/j.compchemeng.2011.12.002. URL: https://www.sciencedirect.com/science/article/pii/S0098135411003395.

[19] C. Loha, H. Chattopadhyay and P. K. Chatterjee. 'Euler-Euler CFD modeling of fluidized bed: Influence of specularity coefficient on hydrodynamic behavior'. In: *Particuology* 11.6 (2013), pp. 673–680. ISSN: 1674-2001. DOI: https://doi.org/10.1016/j.partic.2012.08.007. URL: https://www.sciencedirect.com/science/article/pii/S1674200113000710.

[20] C. Loha, H. Chattopadhyay and P. K. Chatterjee. 'Effect of coefficient of restitution in Euler–Euler CFD simulation of fluidized-bed hydrodynamics'. In: *Particuology* 15 (2014). Energy storage: Materials and processes, pp. 170–177. ISSN: 1674-2001. DOI: https://doi.org/10.1016/j.partic.2013.07.001. URL: https://www.sciencedirect.com/science/article/pii/S1674200113001806.

[21] L. Kong, C. Zhang and J. Zhu. 'Evaluation of the effect of wall boundary conditions on numerical simulations of circulating fluidized beds'. In: *Particuology* 13 (2014), pp. 114–123. ISSN: 1674-2001. DOI: https://doi.org/10.1016/j.partic.2013.04.007. URL: https://www.sciencedirect.com/science/article/pii/S1674200113001417.

[22] C. Altantzis, R.B. Bates and A.F. Ghoniem. '3D Eulerian modeling of thin rectangular gas–solid fluidized beds: Estimation of the specularity coefficient and its effects on bubbling dynamics and circulation times'. In: *Powder Technology* 270 (2015), pp. 256–270. ISSN: 0032-5910. DOI: https://doi.org/10.1016/j.powtec.2014.10.029. URL: https://www.sciencedirect.com/science/article/pii/S0032591014008791.

[23] A. Bakshi et al. 'Eulerian–Eulerian simulation of dense solid–gas cylindrical fluidized beds: Impact of wall boundary condition and drag model on fluidization'. In: *Powder Technology* 277 (2015), pp. 47–62. ISSN: 0032-5910. DOI: https://doi.org/10.1016/j.powtec.2015.02.056. URL: https://www.sciencedirect.com/science/article/pii/S0032591015001898.

[24] H. Shi, A. Komrakova and P. Nikrityuk. 'Fluidized beds modeling: Validation of 2D and 3D simulations against experiments'. In: *Powder Technology* 343 (2019), pp. 479–494. ISSN: 0032-5910. DOI: https://doi.org/10.1016/j.powtec.2018.11.043. URL: https://www.sciencedirect.com/science/article/pii/S0032591018309471.

[25] *Computational Methods for Multiphase Flow*. Cambridge University Press, 2007. DOI: 10.1017/CBO9780511607486.

[26] C. E. Brennen. *Fundamentals of Multiphase Flow*. Cambridge University Press, 2005. DOI: 10.1017/CBO9780511807169.

[27] D. A. Drew and S. L. Passman. *Theory of Multicomponent Fluids*. Springer, 1998. DOI: https://doi.org/10.1007/b97678.

[28] D. Gidaspow. *Multiphase flow and fluidization : continuum and kinetic theory descriptions*. eng. Boston, 1994.

[29] G. Liu. 'Application of the Two-Fluid Model with Kinetic Theory of Granular Flow in Liquid–Solid Fluidized Beds'. In: *Granularity in Materials Science*. Ed. by George Kyzas and Athanasios C. Mitropoulos. Rijeka: IntechOpen, 2018. Chap. 2. DOI: 10.5772/intechopen.79696. URL: https://doi.org/10.5772/intechopen.79696.

[30] N. F. Carnahan and K. E. Starling. 'Equation of State for Nonattracting Rigid Spheres'. In: *The Journal of Chemical Physics* 51.2 (1969), pp. 635–636. DOI: 10.1063/1.1672048. eprint: https://doi.org/10.1063/1.1672048. URL: https://doi.org/10.1063/1.1672048.

[31] C. K. K. Lun et al. 'Kinetic theories for granular flow: inelastic particles in Couette flow and slightly inelastic particles in a general flowfield'. In: *Journal of Fluid Mechanics* 140 (1984), pp. 223–256. DOI: 10.1017/S0022112084000586.

[32] P. C. Johnson and R. Jackson. 'Frictional-collisional constitutive relations for granular materials, with application to plane shearing'. In: *Journal of Fluid Mechanics* 176 (1987), pp. 67–93.

[33] H. D. Young et al. *Sears and Zemansky's University Physics: with Modern Physics*. Thirteenth. San Francisco: Pearson Addison-Wesley, 2012.

[34] J. Ding and D. Gidaspow. 'A Bubbling Fluidization Model Using Kinetic Theory of Granular Flow'. In: *AlChe Journal* 36 (1990), pp. 523–538.

[35] M. Syamlal and T.J. O'Brien. 'Simulation of granular layer inversion in liquid fluidized beds'. In: *International Journal of Multiphase Flow* 14.4 (1988), pp. 473–481. ISSN: 0301-9322. DOI: https://doi.org/10.1016/0301-9322(88)90023-7. URL: https://www.sciencedirect.com/science/article/pii/0301932288900237.

[36] O. Kolditz. 'Numerical Methods'. In: *Computational Methods in Environmental Fluid Mechanics*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2002, pp. 81–96. ISBN: 978-3-662-04761-3. DOI: 10.1007/978-3-662-04761-3_5. URL: https://doi.org/10.1007/978-3-662-04761-3_5.

[37] F. Moukalled, L. Mangani and M. Darwish. 'The Finite Volume Method'. In: *The Finite Volume Method in Computational Fluid Dynamics: An Advanced Introduction with OpenFOAM® and Matlab*. Cham: Springer International Publishing, 2016, pp. 103–135. ISBN: 978-3-319-16874-6. DOI: 10.1007/978-3-319-16874-6_5. URL: https://doi.org/10.1007/978-3-319-16874-6_5.

[38] J. H. Ferziger and Milovan Perić. 'Finite Volume Methods'. In: *Computational Methods for Fluid Dynamics*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2002, pp. 71–89. ISBN: 978-3-642-56026-2. DOI: 10.1007/978-3-642-56026-2_4. URL: https://doi.org/10.1007/978-3-642-56026-2_4.

[39] F. Moukalled, L. Mangani and M. Darwish. 'Temporal Discretization: The Transient Term'. In: *The Finite Volume Method in Computational Fluid Dynamics: An Advanced Introduction with OpenFOAM® and Matlab*. Cham: Springer International Publishing, 2016, pp. 489–533. ISBN: 978-3-319-16874-6. DOI: 10.1007/978-3-319-16874-6_13. URL: https://doi.org/10.1007/978-3-319-16874-6_13.

[40] C. Greenshields and H. Weller. *Computational mesh*. 2022. URL: https://doc.cfd.direct/notes/cfd-general-principles/computational-mesh (visited on 19/05/2023).

[41] H. K. Versteeg and W. Malalasekera. *An Introduction to COMPUTATIONAL FLUID DYNAMICS The Finite Volume Method*. England: Pearson Education, 2007.

[42] D. Goldberg. 'What every computer scientist should know about floating-point arithmetic'. In: *ACM Computing Surveys* 23.1 (1991), pp. 5–48. DOI: 10.1137/0913035. URL: https://doi.org/10.1145/103162.103163.

[43] D. B. Spalding. 'Numerical computation of multi-phase fluid flow and heat transfer'. In: *Recent Advances in Numerical Methods in Fluids*. Vol. 1. 1980, pp. 139–167.

[44] M. Syamlal. *MFIX Documentation Numerical Technique*. Tech. rep. U.S Department of Energy, 1998.

[45] *MFIX*. URL: https://mfix.netl.doe.gov (visited on 21/05/2023).

[46] *OpenFOAM*. URL: https://openfoam.org (visited on 21/05/2023).

[47] P. T. Peeters. 'CFD of multiphase pipe flow: A comparison of solvers'. PhD thesis. TU Delft, 2016.

[48] C. Greenshields and H. Weller. *The PIMPLE algorithm*. 2022. URL: https://doc.cfd.direct/notes/cfd-general-principles/computational-mesh (visited on 21/05/2023).

[49] *About OpenFOAM*. 2023. URL: https://cfd.direct/openfoam/about/ (visited on 21/05/2023).

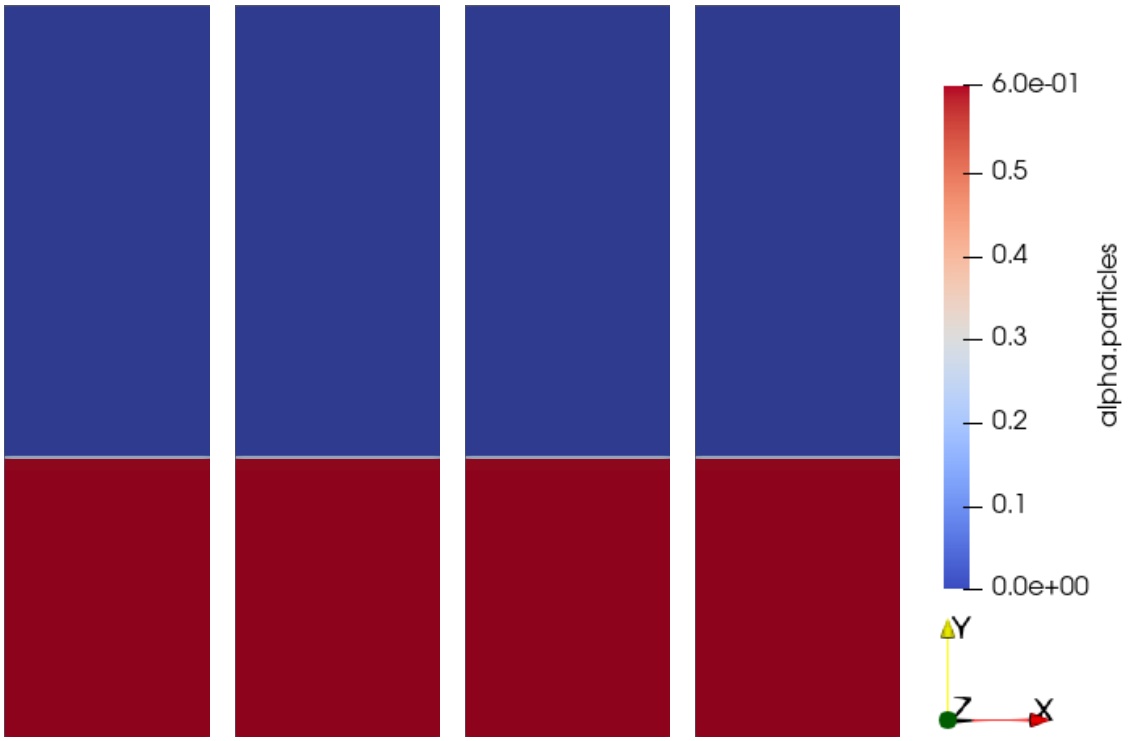
[50] *MFiX User Manual*. 2023. URL: https://mfix.netl.doe.gov/doc/mfix/23.1/html/index.html (visited on 23/05/2023).

Appendix

A Additional velocity results

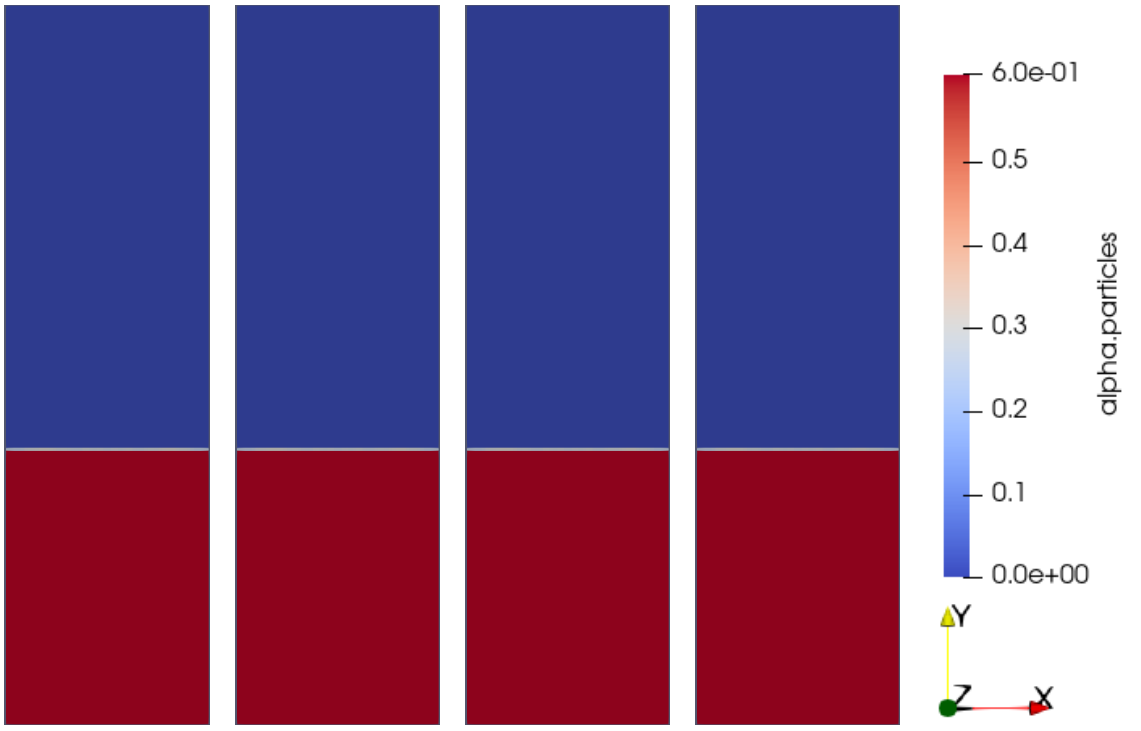
Flow evolution, solid velocity and voidage results for the additional simulations not presented in the results section.

A.1 Superficial gas velocity = 0.03 m/s



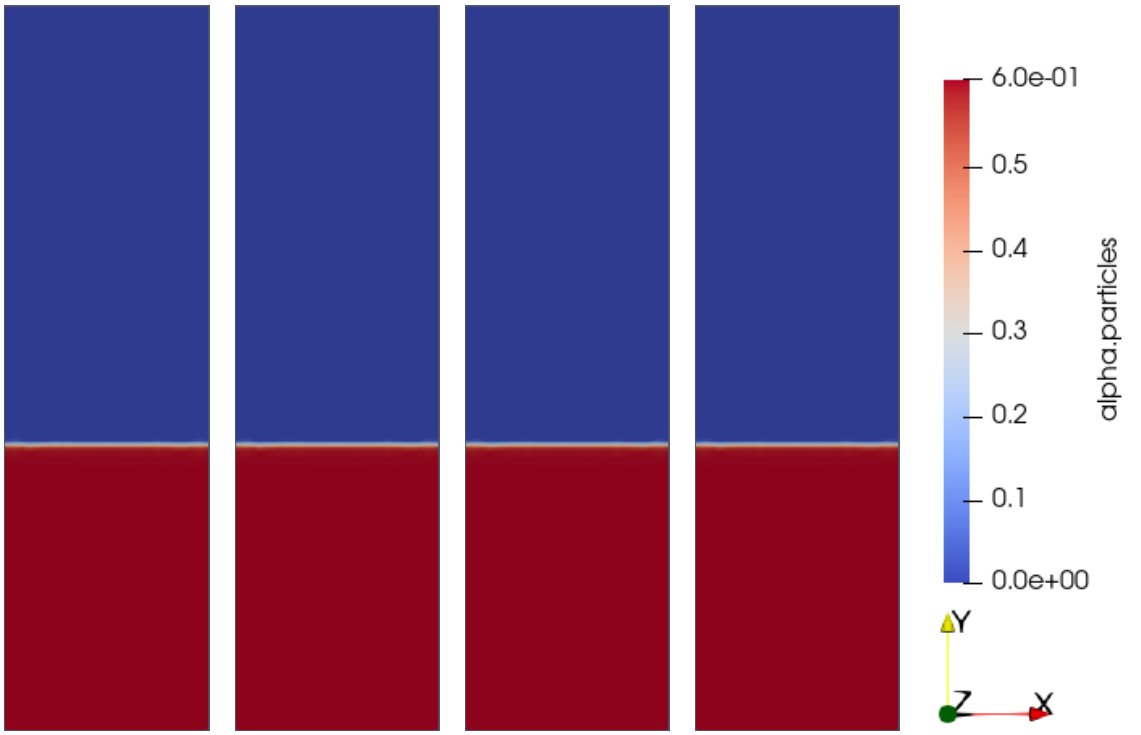
(a) 0.4 seconds (b) 1 second (c) Steady state (d) Time averaged (e) Scale bar

Figure A.1: Particles distribution throughout the simulation with the Syamlal-O'Brien drag model with a superficial gas velocity equal to 0.03 m/s in MFiX.



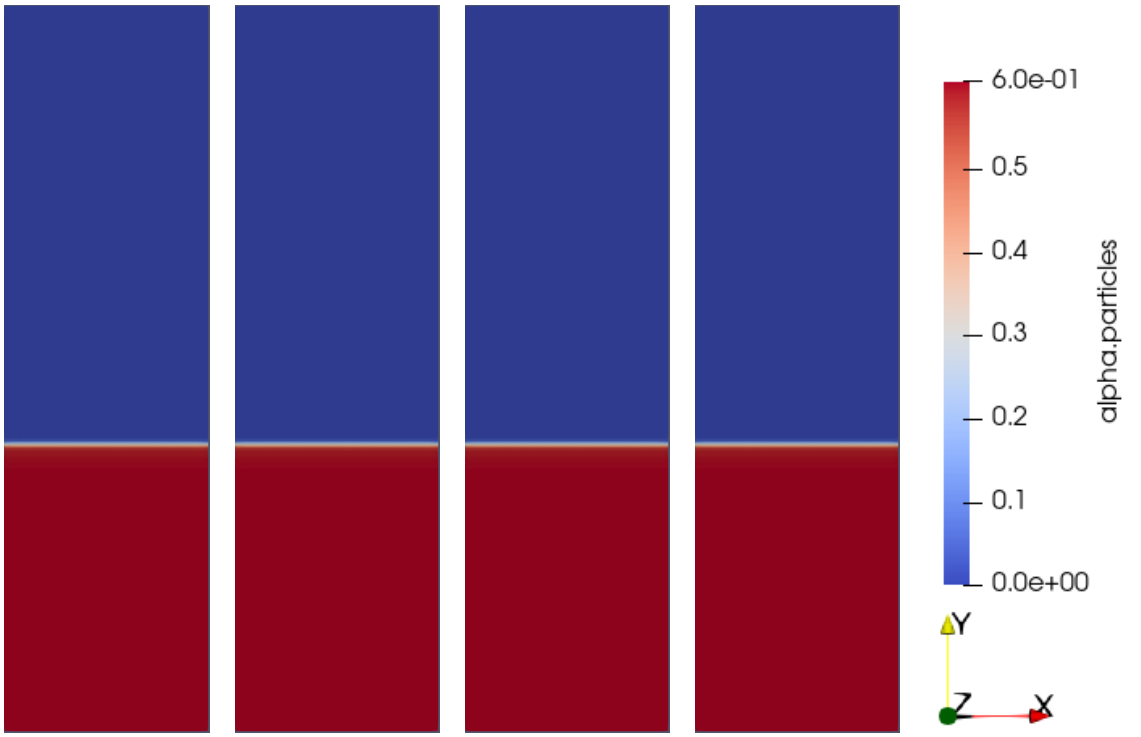
(a) 0.4 seconds (b) 1 second (c) Steady state (d) Time averaged (e) Scale bar

Figure A.2: Particles distribution throughout the simulation with the Gidaspow drag model with a superficial gas velocity equal to 0.03 m/s in MFiX.



(a) 0.4 seconds (b) 1 second (c) Steady state (d) Time averaged (e) Scale bar

Figure A.3: Particles distribution throughout the simulation with the Syamlal-O'Brien drag model with a superficial gas velocity equal to 0.03 m/s in OpenFOAM.



(a) 0.4 seconds (b) 1 second (c) Steady state (d) Time averaged (e) 0.4 seconds

Figure A.4: Particles distribution throughout the simulation with the Gidaspow drag model with a superficial gas velocity equal to 0.03 m/s in OpenFOAM.

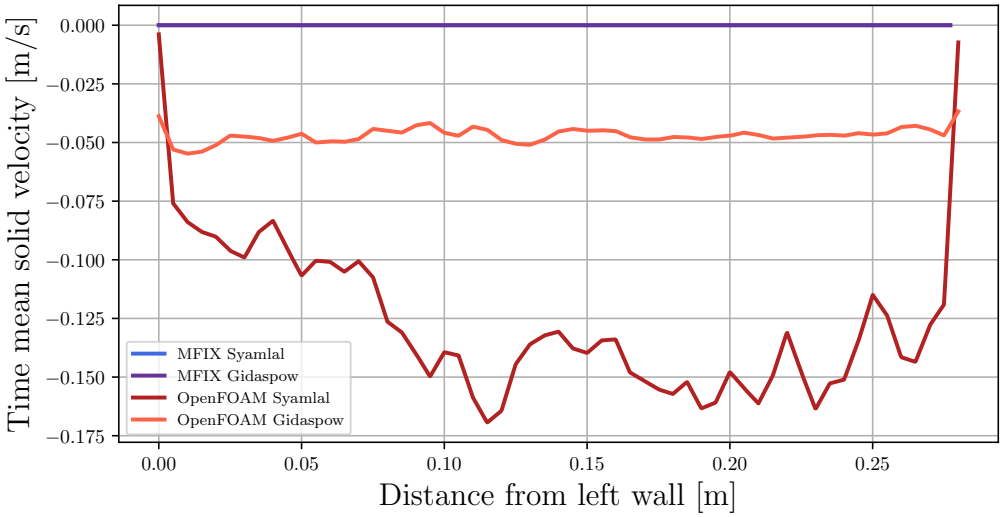


Figure A.5: Time mean solid velocities measured in the range from 3 to 12 seconds with superficial gas velocity equal 0.03 m/s. Measurements taken at y = 0.4 m.

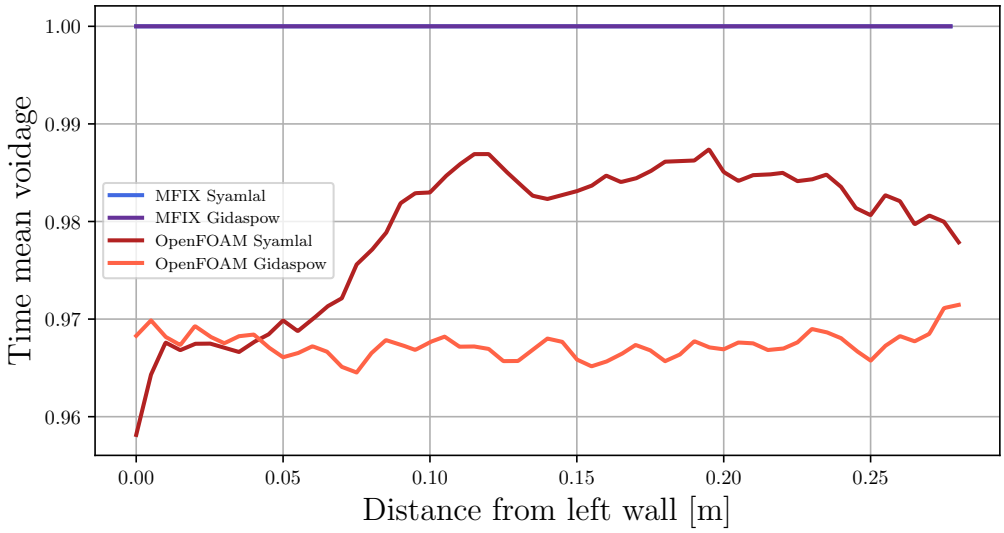
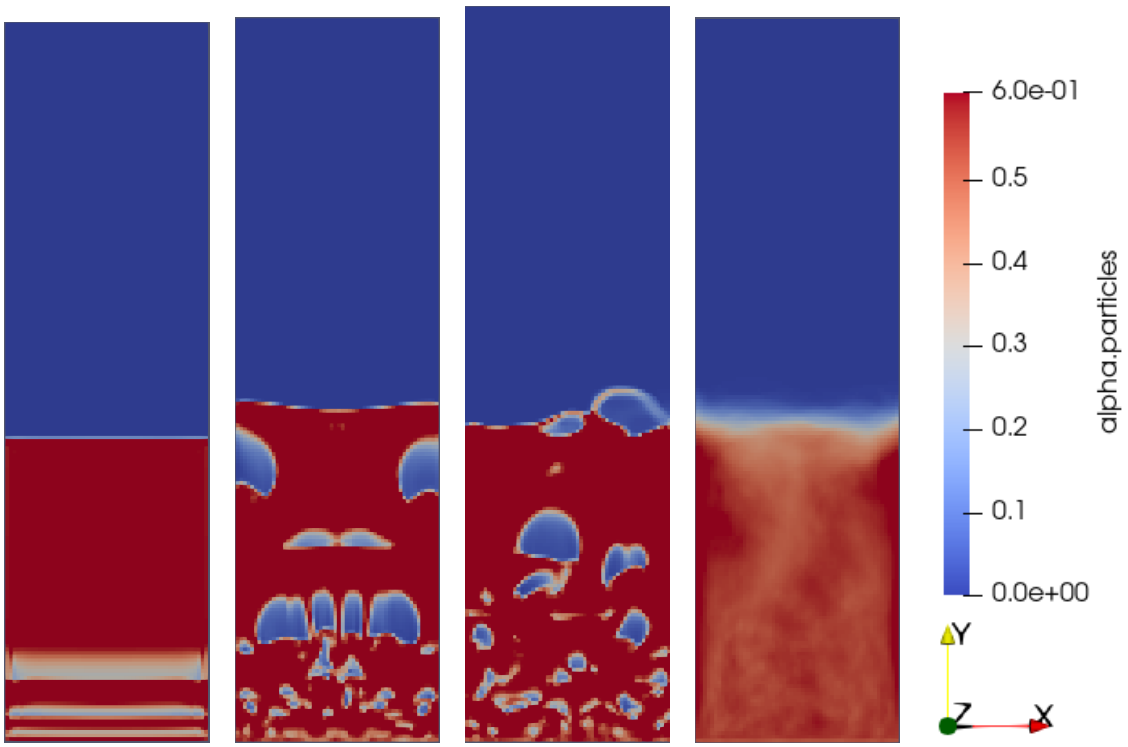


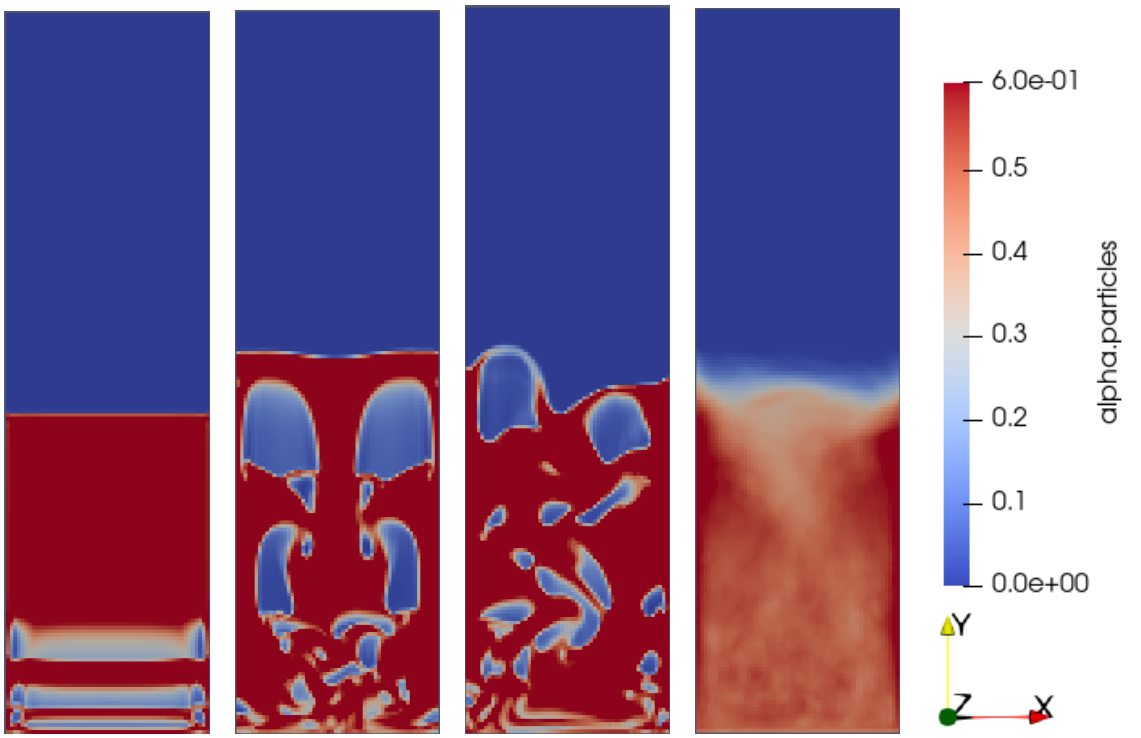
Figure A.6: Time mean voidages measured in the range from 3 to 12 seconds with superficial gas velocity equal 0.03 m/s. The measurements are taken at y = 0.4 m.

A.2 Superficial gas velocity = 0.2 m/s



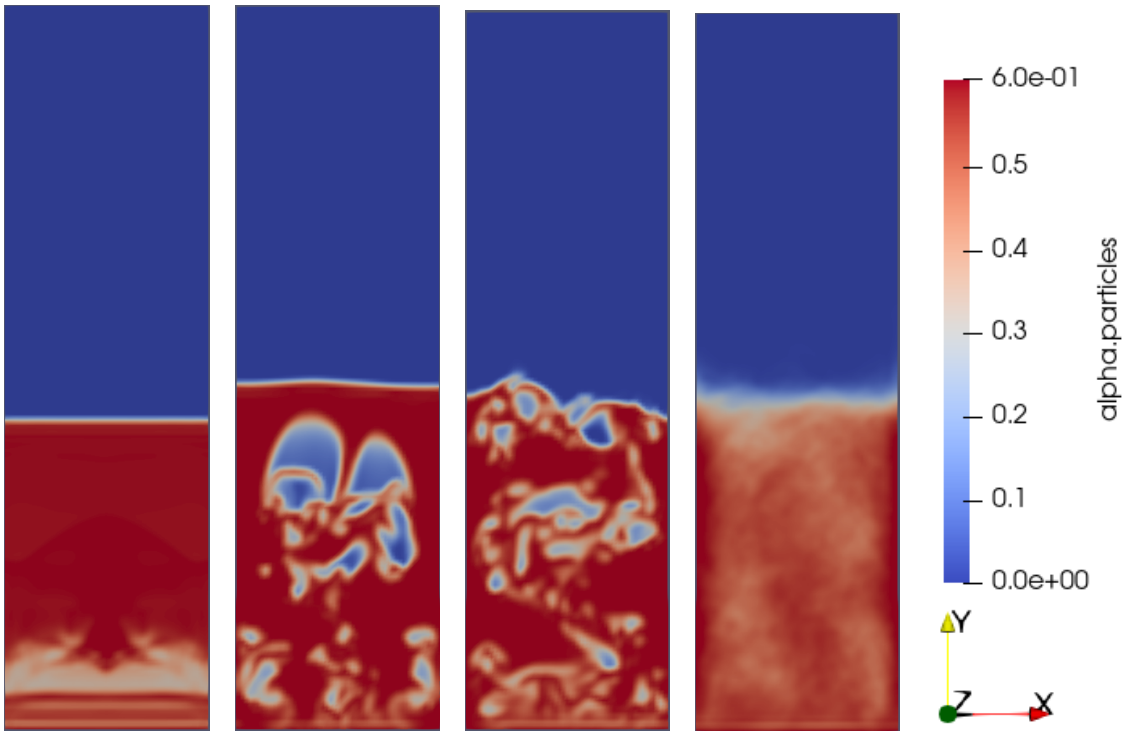
(a) 0.4 seconds (b) 1 second (c) Steady state (d) Time averaged (e) Scale bar

Figure A.7: Particles distribution throughout the simulation with the Syamlal-O'Brien drag model with a superficial gas velocity equal to 0.2 m/s in MFiX.



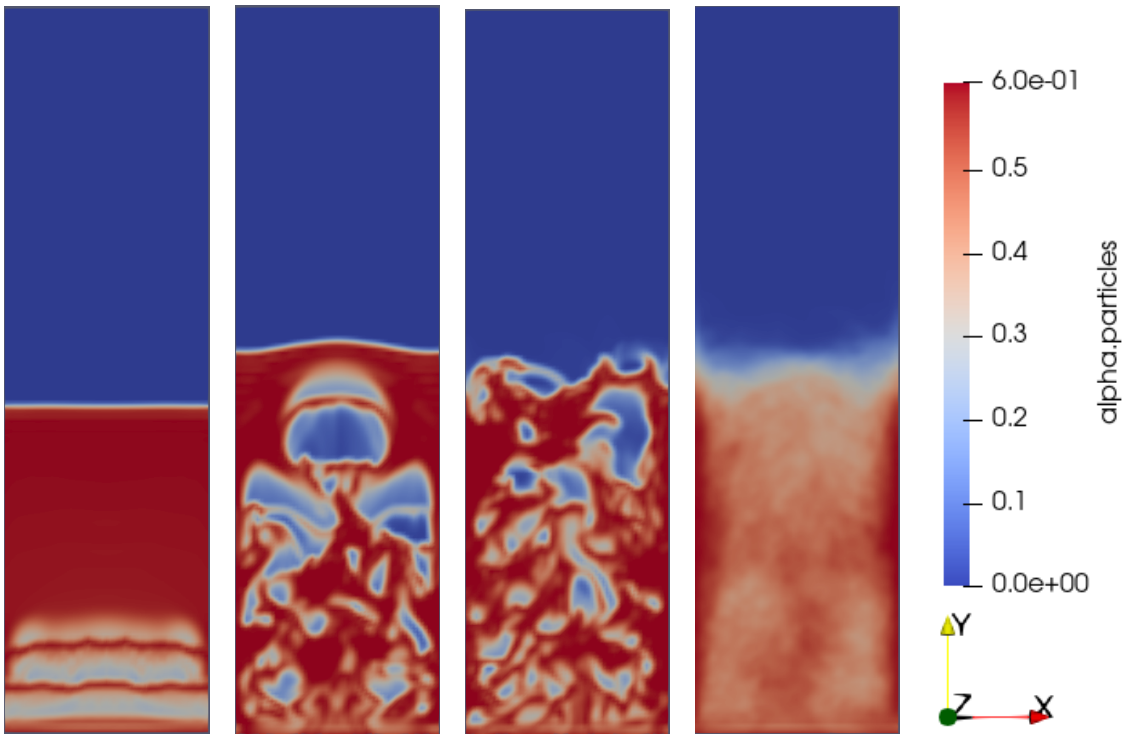
(a) 0.4 seconds (b) 1 second (c) Steady state (d) Time averaged (e) Scale bar

Figure A.8: Particles distribution throughout the simulation with the Gidaspow drag model with a superficial gas velocity equal to 0.2 m/s in MFiX.



(a) 0.4 seconds (b) 1 second (c) Steady state (d) Time averaged (e) Scale bar

Figure A.9: Particles distribution throughout the simulation with the Syamlal-O'Brien drag model with a superficial gas velocity equal to 0.2 m/s in OpenFOAM.



(a) 0.4 seconds (b) 1 second (c) Steady state (d) Time averaged (e) 0.4 seconds

Figure A.10: Particles distribution throughout the simulation with the Gidaspow drag model with a superficial gas velocity equal to 0.2 m/s in OpenFOAM.

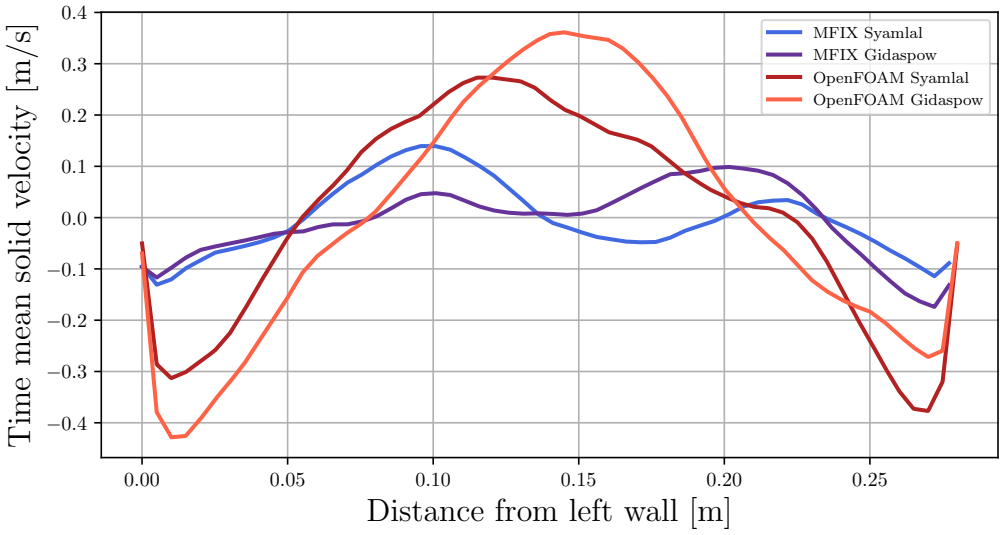


Figure A.11: Time mean solid velocities measured in the range from 3 to 12 seconds with superficial gas velocity equal 0.2 m/s. Measurements taken at y = 0.4 m.

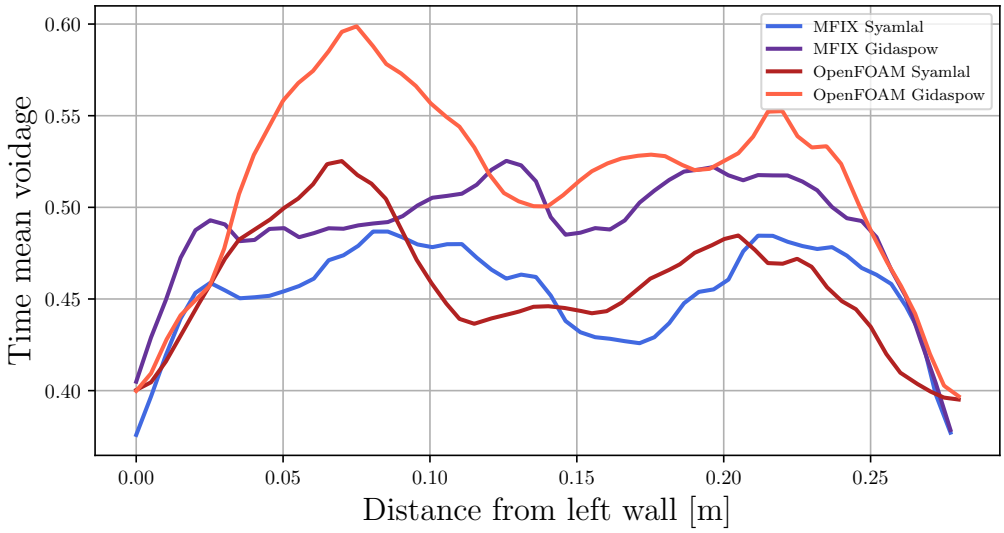
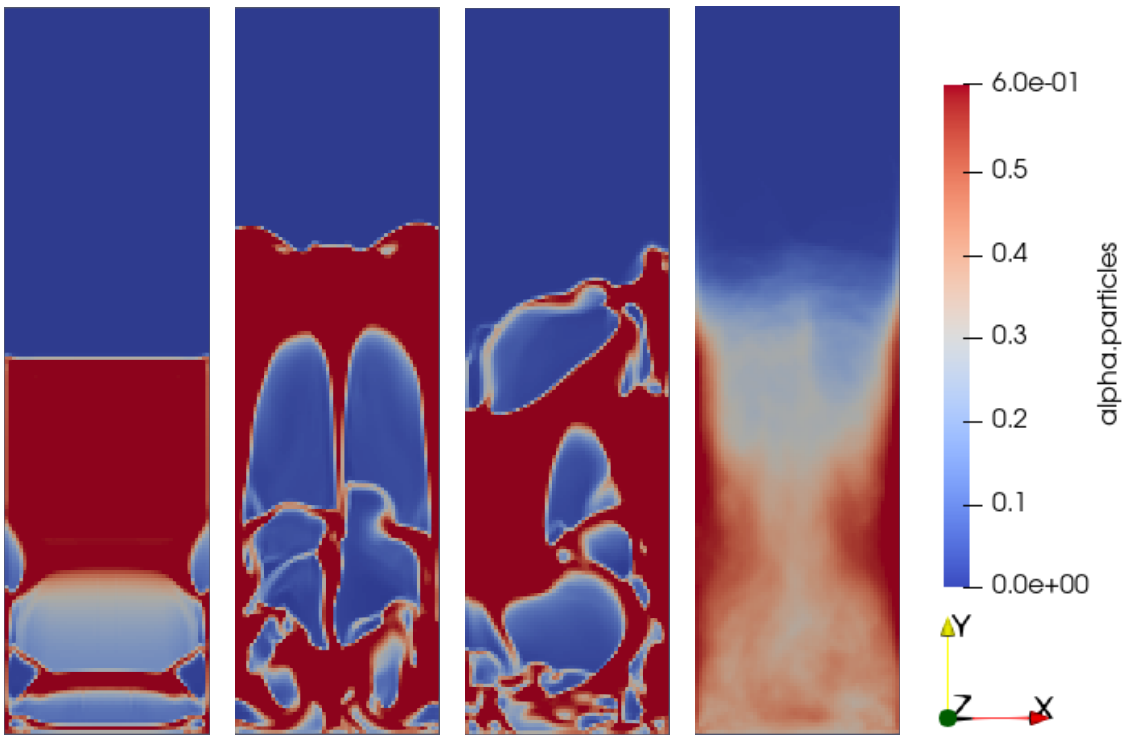


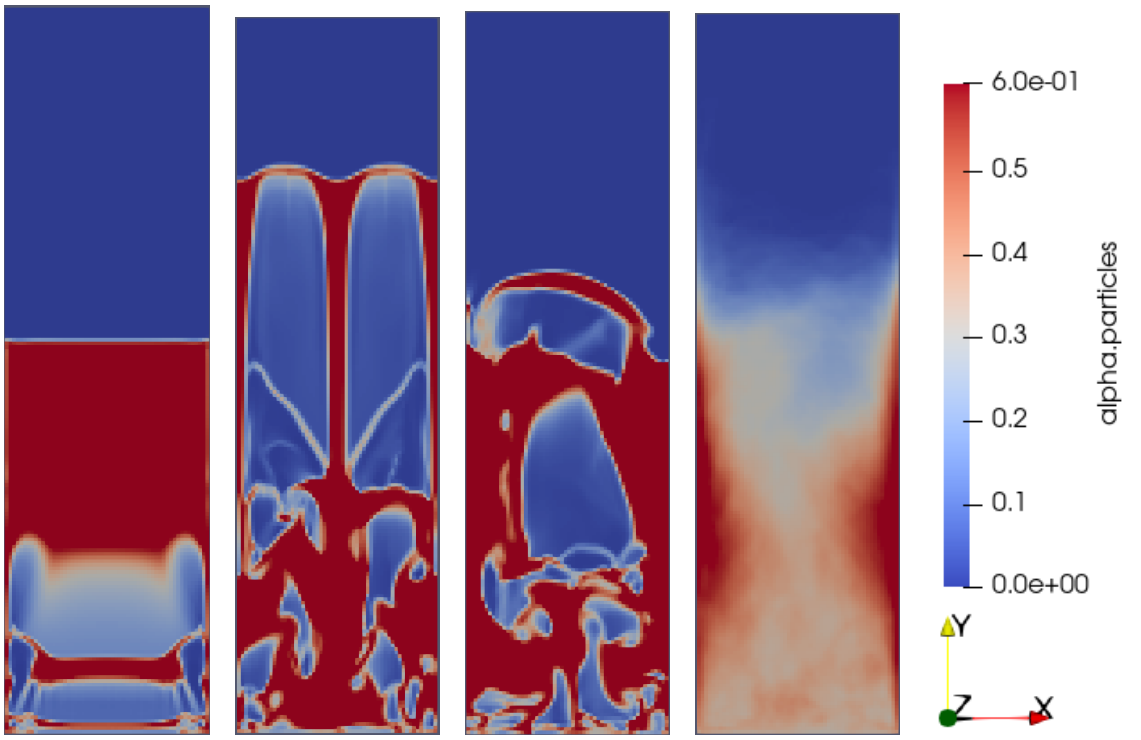
Figure A.12: Time mean voidages measured in the range from 3 to 12 seconds with superficial gas velocity equal 0.2 m/s. The measurements are taken at y = 0.4 m.

A.3 Superficial gas velocity = 0.46 m/s



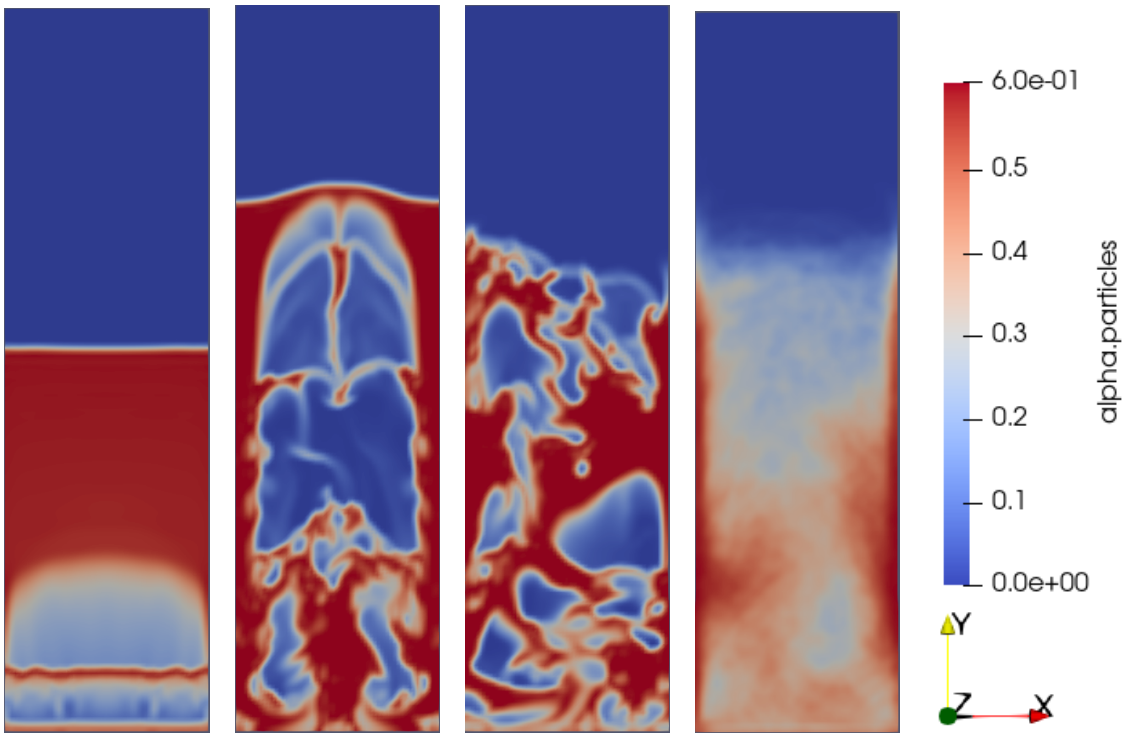
(a) 0.4 seconds (b) 1 second (c) Steady state (d) Time averaged (e) Scale bar

Figure A.13: Particles distribution throughout the simulation with the Syamlal-O'Brien drag model with a superficial gas velocity equal to 0.46 m/s in MFiX.



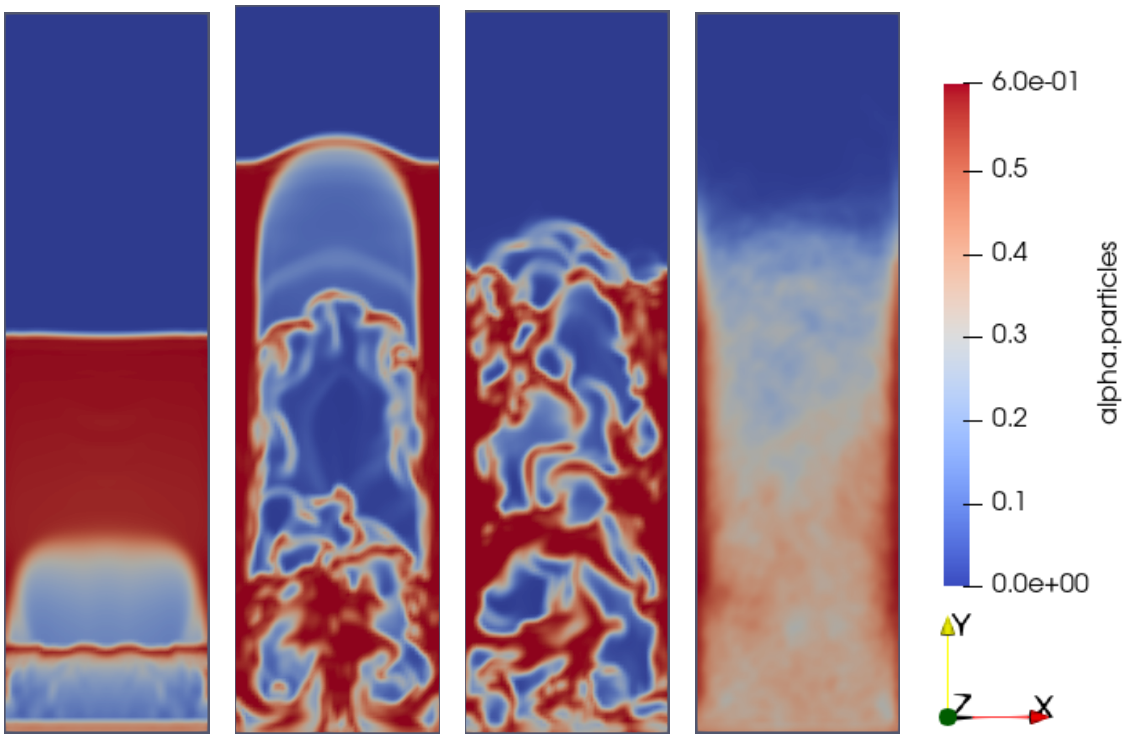
(a) 0.4 seconds (b) 1 second (c) Steady state (d) Time averaged (e) Scale bar

Figure A.14: Particles distribution throughout the simulation with the Gidaspow drag model with a superficial gas velocity equal to 0.46 m/s in MFiX.



(a) 0.4 seconds (b) 1 second (c) Steady state (d) Time averaged (e) Scale bar

Figure A.15: Particles distribution throughout the simulation with the Syamlal-O'Brien drag model with a superficial gas velocity equal to 0.46 m/s in OpenFOAM.



(a) 0.4 seconds (b) 1 second (c) Steady state (d) Time averaged (e) 0.4 seconds

Figure A.16: Particles distribution throughout the simulation with the Gidaspow drag model with a superficial gas velocity equal to 0.46 m/s in OpenFOAM.

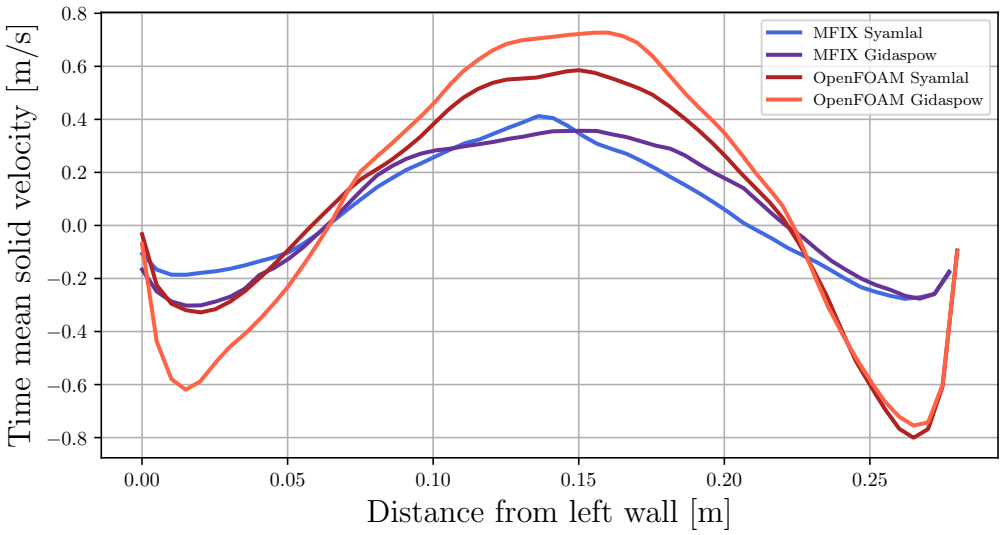


Figure A.17: Time mean solid velocities measured in the range from 3 to 12 seconds with superficial gas velocity equal 0.46 m/s. Measurements taken at y = 0.4 m.

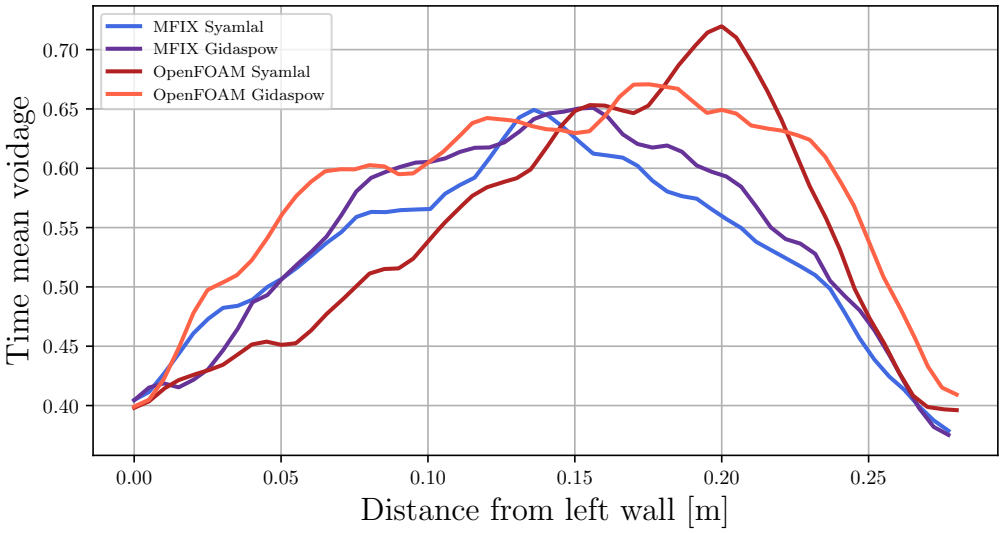


Figure A.18: Time mean voidages measured in the range from 3 to 12 seconds with superficial gas velocity equal 0.46 m/s. The measurements are taken at y = 0.4 m.

B Case setup OpenFOAM

This appendix will cover the entire case setup for a simulation in OpenFOAM. The code listings will not include the OpenFOAM headers and much of the white space is removed to reduce space taken.

B.1 0 directory

```
dimensions      [0 0 0 0 0 0 0];
internalField   uniform 0;
boundaryField
{
    inlet
    {
        type            zeroGradient;
    }
    outlet
    {
        type            zeroGradient;
    }
    walls
    {
        type            zeroGradient;
    }
}
```

Listing B.1: alpha.air.orig

```
dimensions      [0 0 0 0 0 0 0];
internalField   uniform 0;
boundaryField
{
    inlet
    {
        type            zeroGradient;
    }
    outlet
    {
        type            zeroGradient;
    }
    walls
    {
        type            zeroGradient;
    }
}
```

Listing B.2: alpha.particles.orig

```
dimensions      [1 -1 -1 0 0 0 0];
internalField   uniform 0;
boundaryField
{
    inlet
    {
        type            calculated;
        value           $internalField;
    }
    outlet
    {
        type            calculated;
        value           $internalField;
    }
    walls
    {
        type            compressible::alphatWallFunction;
        Prt             0.85;
        value           $internalField;
    }
}
```

Listing B.3: alphat.air

```
dimensions      [1 -1 -1 0 0 0 0];
internalField   uniform 0;
boundaryField
{
    inlet
    {
        type            calculated;
        value           $internalField;
    }
    outlet
    {
        type            calculated;
        value           $internalField;
    }
    walls
    {
        type            calculated;
        value           $internalField;
    }
}
```

Listing B.4: alphat.particles

```
dimensions          [0 2 -3 0 0 0 0];
internalField       uniform 10;
boundaryField
{
    inlet
    {
        type               fixedValue;
        value              $internalField;
    }
    outlet
    {
        type               inletOutlet;
        phi                phi.air;
        inletValue         $internalField;
        value              $internalField;
    }
    walls
    {
        type               epsilonWallFunction;
        value              $internalField;
    }
}
```

Listing B.5: epsilon.air

```
dimensions          [0 2 -2 0 0 0 0];
internalField       uniform 1;
boundaryField
{
    inlet
    {
        type               fixedValue;
        value              $internalField;
    }
    outlet
    {
        type               inletOutlet;
        phi                phi.air;
        inletValue         $internalField;
        value              $internalField;
    }
    walls
    {
        type               kqRWallFunction;
        value              $internalField;
    }
}
```

Listing B.6: k.air

```
dimensions      [0 2 -1 0 0 0 0];
internalField   uniform 0;
boundaryField
{
    inlet
    {
        type            calculated;
        value           $internalField;
    }
    outlet
    {
        type            calculated;
        value           $internalField;
    }
    walls
    {
        type            nutkWallFunction;
        value           $internalField;
    }
}
```

Listing B.7: nut.air

```
dimensions      [0 2 -1 0 0 0 0];
internalField   uniform 0;
boundaryField
{
    inlet
    {
        type            calculated;
        value           $internalField;
    }
    outlet
    {
        type            calculated;
        value           $internalField;
    }
    walls
    {
        type            calculated;
        value           $internalField;
    }
}
```

Listing B.8: nut.particles

```
dimensions      [1 -1 -2 0 0 0 0];
internalField   uniform 101325;
boundaryField
{
    inlet
    {
        type            calculated;
        value           $internalField;
    }
    outlet
    {
        type            calculated;
        value           $internalField;
    }
    walls
    {
        type            calculated;
        value           $internalField;
    }
}
```

Listing B.9: p

```
dimensions      [1 -1 -2 0 0 0 0];
internalField   uniform 101325;
boundaryField
```

```
{
    inlet
    {
        type            fixedFluxPressure;
        value           $internalField;
    }
    outlet
    {
        type            prghPressure;
        p               $internalField;
        value           $internalField;
    }
    walls
    {
        type            fixedFluxPressure;
        value           $internalField;
    }
}
```

Listing B.10: p_rgh

```
dimensions          [0 0 0 1 0 0 0];
internalField       uniform 298;
boundaryField
{
    inlet
    {
        type               fixedValue;
        value              uniform 298;
    }
    outlet
    {
        type               inletOutlet;
        phi                phi.air;
        inletValue         uniform 298;
        value              $internalField;
    }
    walls
    {
        type               zeroGradient;
    }
}
```

Listing B.11: T.air

```
dimensions          [0 0 0 1 0 0 0];
internalField       uniform 298;
boundaryField
{
    inlet
    {
        type               zeroGradient;
    }
    outlet
    {
        type               inletOutlet;
        phi                phi.particles;
        inletValue         uniform 298;
        value              $internalField;;
    }
    walls
    {
        type               zeroGradient;
    }

}
```

Listing B.12: T.particles

```
dimensions          [0 2 -2 0 0 0 0];
internalField       uniform 0;
referenceLevel      1e-4;
```

```
boundaryField
{
    inlet
    {
        type                fixedValue;
        value               uniform 1e-4;
    }
    outlet
    {
        type                zeroGradient;
    }
    walls
    {
        type                      JohnsonJacksonParticleTheta;
        value                     $internalField;
        restitutionCoefficient    0.9;
        specularityCoefficient    0.2;
    }
}
```

Listing B.13: Theta.particles

```
dimensions      [0 1 -1 0 0 0 0];
internalField   uniform (0 0 0);
boundaryField
{
    inlet
    {
        type                interstitialInletVelocity;
        inletVelocity       uniform (0 0.38 0); //0.03, 0.1, 0.2, 0.46, 0.51
        alpha               alpha.air;
        value               $internalField;
    }
    outlet
    {
        type                pressureInletOutletVelocity;
        phi                 phi.air;
        value               $internalField;
    }
    walls
    {
        type            noSlip;
    }
}
```

Listing B.14: U.air

```
dimensions      [0 1 -1 0 0 0 0];
internalField   uniform (0 0 0);
boundaryField
{
    inlet
    {
        type                fixedValue;
        value               uniform (0 0 0);
    }
    outlet
    {
        type                fixedValue;
        value               uniform (0 0 0);
    }
    walls
    {
        type                      JohnsonJacksonParticleSlip;
        value                     $internalField;
        specularityCoefficient    0.2;
    }
}
```

Listing B.15: U.particles

B.2 constant directory

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

Listing B.16: g

```
simulationType  laminar;
```

Listing B.17: momentumTransport.air

```
simulationType  RAS;

RAS
{
    model    kineticTheory;
    turbulence      on;
    printCoeffs     on;
    kineticTheoryCoeffs
    {
        equilibrium             off;
        e                       0.9;
        alphaMinFriction        0.6;
        residualAlpha           1e-4;
        granularViscosityModel      Syamlal; //Gidaspow
        granularConductivityModel   Syamlal; //Gidaspow
        granularPressureModel       Lun;
        frictionalStressModel       Schaeffer;
        radialModel                 CarnahanStarling;
        SchaefferCoeffs
        {
            Fr                      0.05;
            eta                     2;
            p                       5;
            phi                     30;
            alphaDeltaMin           0.05;
        }
    }
    phasePressureCoeffs
    {
        preAlphaExp     500;
        expMax          1000;
        g0              1000;
    }
}
```

Listing B.18: momentumTransport.particles

```
type    basicMultiphaseSystem;

phases (particles air);

referencePhase air;

particles
{
    type          purePhaseModel;
    diameterModel constant;
    constantCoeffs
    {
        d               275e-6;
    }
    alphaMax        0.63;
    residualAlpha   1e-6;
}
air
{
    type          purePhaseModel;
    diameterModel constant;
constantCoeffs
    {
        d               1;
```

```
    }

    residualAlpha    0;
}
blending
{
    default
    {
        type            continuous;
        phase           air;
    }
}
surfaceTension
{}
interfaceCompression
{}
drag
{
    particles_dispersedIn_air
    {
        type            SyamlalOBrien; //GidaspowErgunWenYu
        residualRe      1e-3;
    }
}
virtualMass
{}
heatTransfer
{
    particles_dispersedIn_air
    {
        type            RanzMarshall;
        residualAlpha   1e-3;
    }
}
phaseTransfer
{}
lift
{}
wallLubrication
{}
turbulentDispersion
{}
```

Listing B.19: phaseProperties

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

mixture
{
    specie
    {
        molWeight   28.9;
    }
    thermodynamics
    {
        Cp          1007;
        Hf          0;
    }
    transport
    {
        mu          1.84e-05;
        Pr          0.7;
    }
```

```
}
```

Listing B.20: physicalProperties.air

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

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

Listing B.21: physicalProperties.particles

B.3 system directory

```
convertToMeters 1;
vertices
(
 (0.00 0.0 0.00 )
 (0.28 0.0 0.00 )
 (0.28 1.0 0.00 )
 (0.00 1.0 0.00 )
 (0.00 0.0 0.025)
 (0.28 0.0 0.025)
 (0.28 1.0 0.025)
 (0.00 1.0 0.025)
);
blocks
(
    hex (0 1 2 3 4 5 6 7) (56 200 1) simpleGrading (1 1 1) //(56 200 5) in 3D
);
boundary
(
    inlet
    {
        type patch;
        faces
        (
            (1 5 4 0)
        );
    }
    outlet
    {
        type patch;
        faces
```

```
        (
            (3 7 6 2)
        );
    }
    walls
    {
        type wall;
        faces
        (
            (0 4 7 3)
            (2 6 5 1)
          //(0 3 2 1) //Active here in 3D
            //(4 5 6 7) //Active here in 3D

        );
    }
    frontAndBackPlanes
    {
        type empty;
        faces
        (
            (0 3 2 1) //Active here in 2D
            (4 5 6 7) //Active here in 2D
        );
    }
);
```

Listing B.22: blockMeshDict

```
application     multiphaseEulerFoam;
startFrom       latestTime;
startTime       0;
stopAt          endTime;
endTime         12;
deltaT          1e-4;
writeControl    adjustableRunTime;
writeInterval   0.1;
purgeWrite      0;
writeFormat     binary;
writePrecision  6;
writeCompression off;
timeFormat      general;
timePrecision   6;
runTimeModifiable on;
adjustTimeStep  no;
maxCo           2;
maxDeltaT       0.01;
```

Listing B.23: controlDict

```
limitp
{
    type       limitPressure;
    min        1e4;
}
```

Listing B.24: fvConstraints

```
ddtSchemes
{
    default     Euler;
}
gradSchemes
{
    default     Gauss linear;
}
divSchemes
{
    default                         none;

    "div\(phi,alpha.*\)"            Gauss vanLeer;
    "div\(phir,alpha.*\)"           Gauss vanLeer;
```

```
31
32      "div\(alphaRhoPhi.*,U.*\)"      Gauss limitedLinearV 1;
33      "div\(phi.*,U.*\)"              Gauss limitedLinearV 1;
34
35      "div\(alphaRhoPhi.*,(h|e).*\)"  Gauss limitedLinear 1;
36      "div\(alphaRhoPhi.*,K.*\)"      Gauss limitedLinear 1;
37      "div\(alphaRhoPhi.*,\(p\|thermo:rho.*\)\)" Gauss limitedLinear 1;
38
39      div(alphaRhoPhi.particles,Theta.particles) Gauss limitedLinear 1;
40
41      "div\(alphaRhoPhi.*,(k|epsilon).*\)"  Gauss limitedLinear 1;
42
43      div((((alpha.air*thermo:rho.air)*nuEff.air)*dev2(T(grad(U.air))))) Gauss linear
        ;
44
45      divDevTau(U.particles)          Gauss linear;
46  }
47  laplacianSchemes
48  {
49      default     Gauss linear uncorrected;
50      bounded     Gauss linear uncorrected;
51  }
52  interpolationSchemes
53  {
54      default     linear;
55  }
56  snGradSchemes
57  {
58      default     uncorrected;
59      bounded     uncorrected;
60  }
61  wallDist
62  {
63      method      meshWave;
64  }
```

Listing B.25: fvSchemes

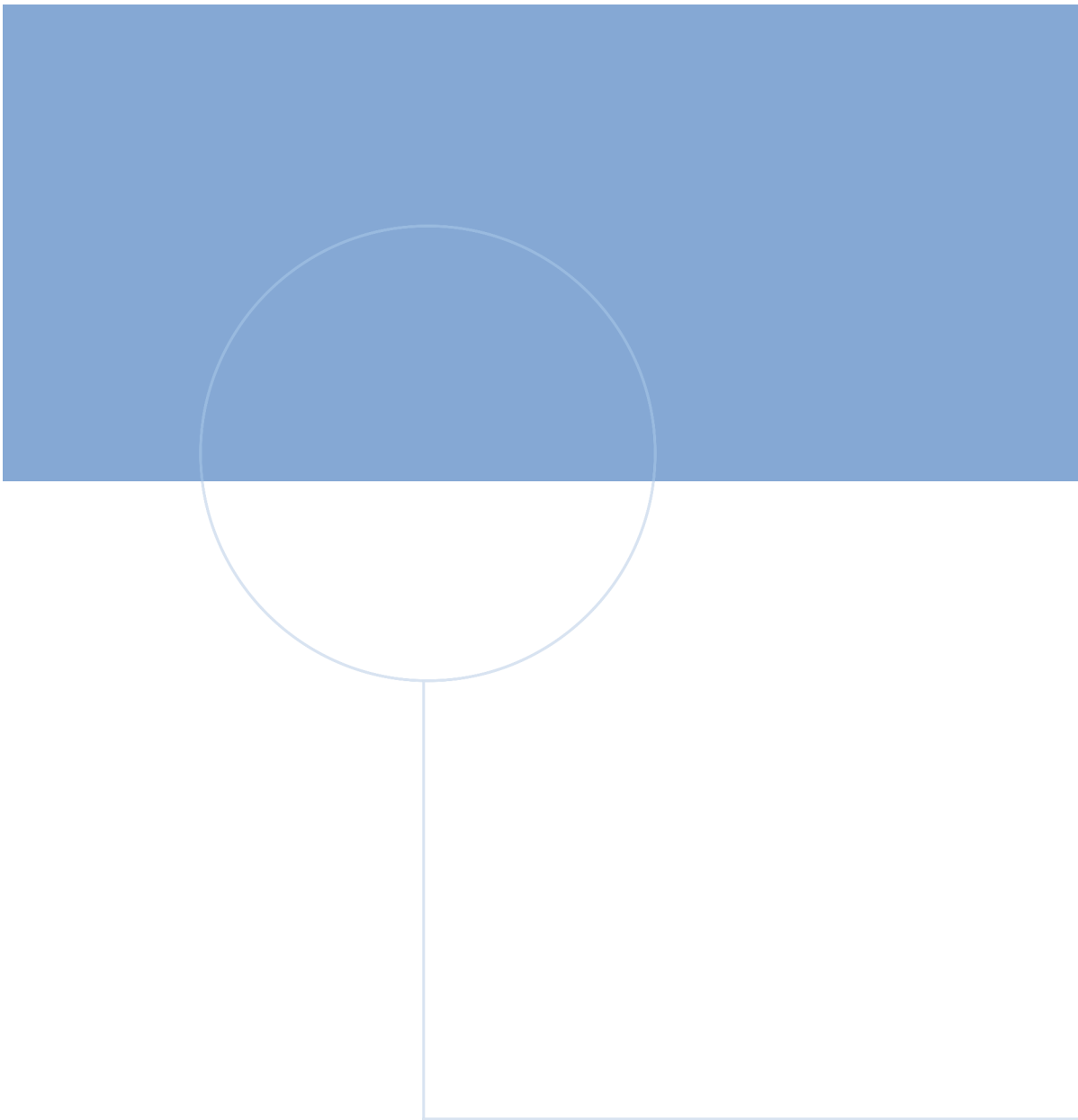
```
17  solvers
18  {
19      "alpha.*"
20      {
21          nAlphaCorr      1;
22          nAlphaSubCycles 3;
23          implicitPhasePressure yes;
24    smoothLimiter 0.1;
25          solver          PBiCGStab;
26    preconditioner DIC;
27          tolerance       1e-9;
28          relTol          0;
29          minIter         1;
30      }
31      p_rgh
32      {
33          solver          GAMG;
34          smoother        DIC;
35          tolerance       1e-8;
36          relTol          0.0;
37      }
38      p_rghFinal
39      {
40          $p_rgh;
41          relTol          0;
42      }
43      "U.*"
44      {
45          solver          smoothSolver;
46          smoother        symGaussSeidel;
47          tolerance       1e-8;
48          relTol          0;
49          minIter         1;
50      }
51      "(h|e).*"
```

```
    {
        solver          smoothSolver;
        smoother        symGaussSeidel;
        tolerance       1e-8;
        relTol          0;
        minIter         1;
        maxIter         10;
    }
    "Theta.*"
    {
         solver          PBiCGStab;
        preconditioner  DILU;
        tolerance       1e-8;
        relTol          0;
        minIter         1;
    }
    "(k|epsilon).*"
    {
        solver          PBiCGStab;
        preconditioner  DILU;
        tolerance       1e-5;
        relTol          0;
        minIter         1;
    }
}
PIMPLE
{
    nOuterCorrectors 3;
    nCorrectors      2;
    nNonOrthogonalCorrectors 0;
    faceMomentum     no;
}
relaxationFactors
{
    equations
    {
        ".*"            1;
    }
}
```

Listing B.26: fvSolution

```
defaultFieldValues
(
    volScalarFieldValue alpha.air 1
    volScalarFieldValue alpha.particles 0
);
regions
(
    boxToCell
    {
        box (0 0 0) (0.28 0.4 0.025);
        fieldValues
        (
            volScalarFieldValue alpha.air 0.4
            volScalarFieldValue alpha.particles 0.6
        );
    }
);
```

Listing B.27: setFieldsDict



NTNU

Norwegian University of
Science and Technology