Parallelization of a stochastic Euler-Lagrange model applied to large scale dense bubbly flows

Kamath, S.; Masterov, M. V.; Padding, J. T.; Buist, K. A.; Baltussen, M. W.; Kuipers, J. A.M.

DOI
10.1016/j.jcpx.2020.100058

Publication date
2020

Document Version
Final published version

Published in
Journal of Computational Physics: X

Citation (APA)
Kamath, S., Masterov, M. V., Padding, J. T., Buist, K. A., Baltussen, M. W., & Kuipers, J. A. M. (2020). Parallelization of a stochastic Euler-Lagrange model applied to large scale dense bubbly flows. *Journal of Computational Physics: X*, *8*, Article 100058. https://doi.org/10.1016/j.jcpx.2020.100058

Journal of Computational Physics: X 8 (2020) 100058

Contents lists available at ScienceDirect

Journal of Computational Physics: X

www.elsevier.com/locate/jcpx

Parallelization of a stochastic Euler-Lagrange model applied to large scale dense bubbly flows

S. Kamath [a,1], M.V. Masterov [a,1], J.T. Padding [b,*], K.A. Buist [a,*], M.W. Baltussen [a], J.A.M. Kuipers [a]

[a] Multiphase Reactors Group, Department of Chemical Engineering & Chemistry, Eindhoven University of Technology, P.O. Box 513, 5600 MB Eindhoven, the Netherlands
[b] Process and Energy Department, Delft University of Technology, Leeghwaterstraat 39, 2628 CB Delft, the Netherlands

ARTICLE INFO

Article history:
Received 3 October 2019
Received in revised form 2 April 2020
Accepted 18 April 2020
Available online 4 May 2020

Keywords:
Parallelization
Bubble columns
Euler-Lagrange modelling
Direct Simulation Monte Carlo

ABSTRACT

A parallel and scalable stochastic Direct Simulation Monte Carlo (DSMC) method applied to large-scale dense bubbly flows is reported in this paper. The DSMC method is applied to speed up the bubble-bubble collision handling relative to the Discrete Bubble Model proposed by Darmana et al. (2006) [1]. The DSMC algorithm has been modified and extended to account for bubble-bubble interactions arising due to uncorrelated and correlated bubble velocities. The algorithm is fully coupled with an in-house CFD code and parallelized using the MPI framework. The model is verified and validated on multiple cores with different test cases, ranging from impinging particle streams to laboratory-scale bubble columns. The parallel performance is shown using two different large scale systems: with an uniform and a non-uniform distribution of bubbles. The hydrodynamics of a pilot-scale bubble column is analyzed and the effect of the column scale is reported via the comparison of bubble columns at three different scales.

1. Introduction

Bubbly flows are one of the more complex multi-scale multiphase flow problems which are encountered across many fields of physics and engineering. Small scale physical changes in the properties of the different phases can have a huge impact on the large scale physics of systems. Such flows are studied at the DNS (Direct Numerical Simulations) level for representative systems to derive closures for interactions forces used in simulations executed at larger length and time scales. Methods such as the Discrete Bubble Model (DBM) and also the stochastic Euler-Lagrange (E-L) models (among which DSMC methods) are examples of classes of methods where these closures are applied. These models (DBM/DSMC) have been extensively used across many length and time scales for the simulation of multi-phase flows and have proven to provide good results on relatively coarse grids compared to fully resolved DNS, while remaining more accurate than the further coarse grained Euler-Euler (E-E) models.

However, due to the large amount of individually tracked objects in large-scale systems, the application of Lagrangian Discrete Methods (LDM) in multi-phase flows is complicated due to the high computational effort and memory requirements. The limitations can be overcome with effective parallelization of the numerical code, which can be realized using

* Corresponding authors.
E-mail addresses: j.t.padding@tudelft.nl (J.T. Padding), k.a.buist@tue.nl (K.A. Buist).
[1] Authors S. Kamath and M.V. Masterov have contributed equally to this work.

https://doi.org/10.1016/j.jcpx.2020.100058

Table 1
List of recent publications on parallel implementations of Lagrangian methods.

Name	Type	CFD grid size	#Lagr. elements	Parallelization technique	#Cores	Efficiency
Ouro et al. [2]	CFD+IBM	$52.59 \cdot 10^6$	$3.57 \cdot 10^5$	MPI+OpenMP	314	ideal
Rakotonirinaa et al. [3]	CFD+DEM	-	$2.304 \cdot 10^8$	MPI	768	90%
Pozzetti et al. [4]	CFD+DEM	10^6	10^7	MPI	280	70%
Tian et al. [5]	DEM	-	$1.28 \cdot 10^8$	GPU/MPI	256/128	62.59%/20%
Spandan et al. [6]	CFD+IBM	$2 \cdot 10^9$	$2.5 \cdot 10^4$	MPI	1000	78%
Wang et al. [7]	CFD+DEM	$3 \cdot 10^5$	$2 \cdot 10^5$	MPI	64	57.5%
Geneva et al. [8]	CFD+IBM	$1.6 \cdot 10^7$	7% vol. frac.	MPI	580	ideal
Loisy et al. [9]	CFD+DEM	$1.68 \cdot 10^7$	8	MPI	512	90%
Liu et al. [10]	CFD+DEM	$1.25 \cdot 10^5$	10^4	MPI	1000	50%
Yue et al. [11]	DEM	$2 \cdot 10^4$	8	GPU	2880	x20 speedup
Mountrakis et al. [12]	LBM+IBM	$1.342 \cdot 10^8$	$6.624 \cdot 10^5$	MPI	8192	79%
Ma et al. [13]	DBM	$6.5 \cdot 10^4$	$1.7 \cdot 10^5$	OpenMP	12	99%
Goniva et al. [14]	CFD+DEM	$2.05 \cdot 10^5$	$2.05 \cdot 10^7$	MPI	512	>100%
Berger et al. [15]	DEM	-	$5 \cdot 10^4$	MPI+OpenMP	128	50%
Wu et al. [16]	CFD+DEM	$1.28 \cdot 10^5$	$2 \cdot 10^6$	MPI+OpenMP	128	60%/20%
Niethammer et al. [17]	MD	-	10^{12}	MPI+OpenMP	65536	82.5%
Liu et al. [18]	CFD+DEM	$1.6 \cdot 10^6$	$5.12 \cdot 10^6$	MPI+OpenMP	256/128	37.5%
Amritkar et al. [19]	CFD+DEM	-	$1.05 \cdot 10^6$	MPI+OpenMP	64	-
Kuan et al. [20]	CFD+FT	$6.4 \cdot 10^4$	-	MPI	128	20%
Kafui et al. [21]	CFD+DEM	-	$5 \cdot 10^4$	MPI	64	58%
Darmana et al. [1]	CFD+DBM	$6.5 \cdot 10^6$	10^5	MPI	32	50%
Maknickas et al. [22]	DEM	-	10^5	MPI	16	70%
Pohl et al. [23]	LBM	$1.078 \cdot 10^9$	-	MPI	512	75%

stand alone and combined models for distributed and shared memory platforms, such as: Message Passing Interface (MPI), OpenMP and programming on Graphics Processing Units (GPU). For an overview of the recent parallelization strategies applied to LDM methods see Table 1.

The main drawback of MPI-based codes is in the necessity of keeping an optimal load balance between distributed processes, which is often hard or even impossible to reach in a real case scenario. For instance, in the Euler-Lagrange (E-L) framework, a non-even distribution of the dispersed phase may lead to reduction of the parallel performance or even idling of some processes during execution of the Lagrangian part. This problem is mainly caused by the static domain decomposition based on the Eulerian grid. The obvious solution is to use a separate dynamic decomposition for the Lagrangian part. The coupling between two numerical domains can be achieved by means of a map, that should be reconstructed during the update of the Lagrangian domain decomposition. This approach was successfully applied by Pozzetti et al. [24], who reported an excellent parallel efficiency of the E-L framework up to 1400 cores. However, in a uniformly distributed system, this approach may also lead to extra communications and, therefore, to a decrease in parallel efficiency.

The computational and parallel efficiency of LDMs highly depends on the type of collision detection method employed in the algorithm. Particle based simulations, which often utilize a soft-sphere or a force based collision model, employ a single loop-based algorithm, thus leading to optimal parallel performance on any platform, as shown by Niethammer et al. [17], Pohl et al. [23] and Tian et al. [5], who performed simulation of systems containing 10^{12}, 10^9 and 10^8 particles, respectively. Most of these works are based on the decomposition of only Lagrangian entities in the domain, which can easily incorporate load-balancing with the latest state of the art algorithms.

Historically, bubble-based simulations have often employed a hard-sphere model, which is an event-driven algorithm that requires calculation of the shortest collision time between all possible pairs of bubbles. The efficient parallelization of the DBM model on distributed memory platforms is impossible due to the necessity of continuous synchronization between all processes, while shared memory platforms limit the amount of bubbles that can be simulated. Several recent works also reported the use of soft sphere models for bubbly flow simulations. However, these soft sphere models generally require determination of a spring constant which defines the amount of overlap that is allowed. This is challenging since a high spring constant leads to very small time steps, while a too low spring constant leads to significant over-packing. To avoid spurious velocities in dense systems modelled with a soft-sphere model, very high spring constants and, therefore, extremely low time-steps are required to resolve the collisions sufficiently accurate. Therefore, the number of works conducted on parallelization of bubbly flows at this scale is very limited.

Darmana et al. [25] simulated $\mathcal{O}(10^5)$ bubbles using a mirror domain technique for the Discrete Bubble Model (DBM) combined with an MPI-based CFD code. These authors reported only 50% of parallel efficiency on 32 cores. Ma et al. [13] simulated a bubble column with $6.5 \cdot 10^4$ bubbles using a DBM code parallelized with OpenMP and reported 99% parallel efficiency on 12 threads. Sungkorn et al. [26] have reported a 40 m^3 stirred bio-reactor with about 15.6 million bubbles using a LBM based flow-solver and the bubble phase being solved on GPUs. This approach used a ghost-particle technique developed by Sommerfeld et al. [27]. However, the volume displacement of the bubbles is not accounted for in the flow solver via the volume fraction term and is only coupled through the volumetric source term in the momentum equations. Recently, Kamath et al. [28] have reported a DSMC based stochastic Euler-Lagrange method which accounts for the volume

Table 2
Closures for the different types of forces acting on bubbles.

Force	Closure	Reference
$\mathbf{F}_G = \rho_b V_b \mathbf{g}$	–	–
$\mathbf{F}_p = -V_b \nabla P$	–	–
$\mathbf{F}_D = -\frac{1}{2} C_D \rho_l \pi R_b^2 \lvert\mathbf{v}-\mathbf{u}\rvert(\mathbf{v}-\mathbf{u})$	$C_{D,\infty} = \max\left[\min\left\{\frac{16}{Re}(1+0.15Re^{0.687}), \frac{48}{Re}\right\}, \frac{8}{3}\frac{E\ddot{o}}{E\ddot{o}+4}\right]$	
	$\frac{C_D}{C_{D,\infty}(1-\varepsilon_b)} = 1 + \frac{18}{E\ddot{o}}\varepsilon_b$	[29–31]
$\mathbf{F}_L = -C_L \rho_l V_b (\mathbf{v}-\mathbf{u}) \times (\nabla \times \mathbf{u})$	$C_L = \begin{cases} \min[0.288\tanh(0.121Re), f(E\ddot{o}_d)], & E\ddot{o}_d < 4 \\ f(E\ddot{o}_d), & 4 \le E\ddot{o}_d \le 10 \\ -0.29, & E\ddot{o}_d > 10 \end{cases}$	
	$f(E\ddot{o}_d) = 0:00105E\ddot{o}_d^3 - 0.0159E\ddot{o}_d^2 - 0.0204E\ddot{o}_d + 0.474$	
	$E\ddot{o}_d = \frac{E\ddot{o}}{E};\ E = \frac{1}{1+0.136E\ddot{o}^{0.757}}$	[29,30]
$\mathbf{F}_{VM} = -C_{VM}\rho_l V_b\left(\frac{D_b\mathbf{v}}{D_b t} - \frac{D_b\mathbf{u}}{D_b t}\right)$	$C_{VM} = 0.5$	[25]
$\mathbf{F}_W = C_W R_b \rho_l V_b \frac{1}{D_{bw}^2}\lvert\mathbf{u}-\mathbf{v}\rvert^2 \mathbf{n}_W$	$C_W = \begin{cases} \exp(-0.933E\ddot{o}+0.179), & 1 \le E\ddot{o}_d < 5 \\ 0.0007E\ddot{o}+0.04, & 5 < E\ddot{o}_d \ge 33 \\ 0.179, & E\ddot{o}_d > 33 \end{cases}$	[29,30]

displacement of the bubbles in the Eulerian phase and also has the potential to be parallellized for large scale bubbly flow simulations.

This paper presents an efficient highly scalable parallel approach that allows one to simulate dense bubbly flows in a Lagrangian formulation using the MPI framework. The bubble collisions are resolved by means of Direct Simulation Monte Carlo (DSMC), proposed earlier by Kamath et al. [28]. To the authors' knowledge, this is the first time a DSMC based method is used to simulate dense bubbly flows at this scale. The DSMC operates with a collision probability that can be calculated locally and thus the method does not require global synchronizations and demonstrates high parallel efficiency. As a result, the fully coupled CFD and DSMC code can be applied to simulations of large scale bubble columns with more than 10^6 bubbles.

The paper is structured as follows. Section 2 describes the DSMC method applied to bubbles and the governing equations for the liquid phase. In section 3, the numerical approach for both phases and their coupling is discussed. The parallelization strategy for the gas and liquid phases is described in section 4. In section 5, the verification and validation of the parallel code is reported. Section 6 is dedicated to the investigation of the parallel performance. In section 7.1 an example of a simulation of a large scale bubble column is shown. Finally, the conclusions are presented in section 8.

2. Methodology

2.1. Discrete phase

The discrete phase or bubble phase is solved in a Lagrangian framework where the bubble motion is governed by Newton's equation of motion (equation (1)). The interaction forces included in this model are the gravitational ($\mathbf{F}_G$), pressure ($\mathbf{F}_P$), drag ($\mathbf{F}_D$), lift ($\mathbf{F}_L$), virtual mass ($\mathbf{F}_{VM}$) and wall lubrication ($\mathbf{F}_W$) force (equation (2)). The relevant closures used for these forces can be found in Table 2.

$$\rho_b V_b \frac{d\mathbf{v}}{dt} = \Sigma\mathbf{F} - (\rho_b \frac{d(V_b)}{dt})\mathbf{v} \tag{1}$$

$$\Sigma\mathbf{F} = \mathbf{F}_G + \mathbf{F}_P + \mathbf{F}_D + \mathbf{F}_L + \mathbf{F}_{VM} + \mathbf{F}_W \tag{2}$$

The DSMC methods generally employ a parcel approach where a group of discrete particles of the same size and velocity is represented by one simulated particle. The resolution of the DSMC method is also governed by the number of discrete phase entities per parcel. In the current work, the parcel size for all simulations is taken as 1, to obtain an accurate representation of the Lagrangian phase. Moreover, the extension to larger parcel sizes requires closures determined with careful consideration of the different forces acting on the bubbles as a swarm and mapping functions for the calculation of the porosity. This is caused by the non-linear trajectory of the bubbles due to coupling with the liquid. The parcel growth cannot be assumed to be isotropic. Moreover, parcels of similar size should be able to break-up and coalesce based on the fluid-parcel interactions and the induced turbulence in the liquid. This behaviour is complex due to the four way coupling and this aspect will be considered in our future work.

The main goal of the DSMC method is to predict the frequency and probability of collisions of particles. This stochastic approach is much more efficient than deterministic collision models. The method was initially developed and applied to molecular systems that have finite/large Knudsen number with no external force fields affecting the position and velocity

of the parcels [32]. The modified Nanbu method is a DSMC type method used by Pawar et al. [33] to simulate small droplets in spray modelling with a few alterations. Two modifications are made to the algorithm reported by Pawar et al. [33] by Kamath et al. [28]. First of all, the radial distribution function at contact is added to account for the increase in the collision frequency and the probability of collisions for dense clusters. Secondly, a nearest neighbour collision check is added accounting for discrete phase entities with correlated velocities which are not accounted for by the previously mentioned algorithms.

2.2. Liquid phase hydrodynamics

The liquid dynamics are resolved by means of the volume-averaged Navier-Stokes equations:

$$\frac{\partial \varepsilon_l}{\partial t}\Omega + \int_\sigma \varepsilon_l \mathbf{u} \cdot d\mathbf{A} = 0 \tag{3}$$

$$\rho_l \frac{\partial \varepsilon_l \mathbf{u}}{\partial t}\Omega + \rho_l \int_\sigma \varepsilon_l \mathbf{u}\mathbf{u} \cdot d\mathbf{A} = -\varepsilon_l \nabla p \Omega - \int_\sigma \varepsilon_l \boldsymbol{\tau}_l \cdot d\mathbf{A} + \varepsilon_l \rho_l \mathbf{g}\Omega + \Phi\Omega \tag{4}$$

where Ω is the volume of the grid cell, A is the face area of the grid cell, ε_l is the local liquid volume fraction, Φ represents an external source term due to the presence of the gas phase and $\boldsymbol{\tau}_l$ is the viscous stress tensor given by:

$$\boldsymbol{\tau}_l = -\mu_{eff,l}\left[(\nabla \mathbf{u}) + (\nabla \mathbf{u})^T - \frac{2}{3}\mathbf{I}(\nabla \cdot \mathbf{u})\right] \tag{5}$$

where

$$\mu_{eff,l} = \mu_{L,l} + \mu_{T,l} \tag{6}$$

The $\mu_{L,l}$ and $\mu_{T,l}$ are the dynamic molecular and eddy viscosity, respectively. To close the system we use the subgrid scale model proposed by Vreman et al. [34]:

$$\mu_{T,l} = \rho_l 2.5 C_s^2 \sqrt{\frac{\beta_b}{\alpha_{ij}\alpha_{ij}}}, \quad \alpha_{ij} = \frac{\partial u_j}{\partial x_i}$$
$$\beta_b = \beta_{11}\beta_{22} - \beta_{12}^2 + \beta_{11}\beta_{33} - \beta_{13}^2 + \beta_{22}\beta_{33} - \beta_{23}^2, \quad \beta_{ij} = \Omega^{2/3}\alpha_{mi}\alpha_{mj} \tag{7}$$

where C_s is the Smagorinsky constant equal to 0.1.

3. Numerical solution method

3.1. Time marching

The numerical solution of the governing equations for our Euler-Lagrange model is obtained using a time-marching technique similar to the one reported in Darmana et al. [25] where the equations for the discrete and continuous phase are solved sequentially, while accounting for the coupling via the momentum source terms. Three time steps are defined in this model: the flow time-step (δt_{flow}), the Lagrangian phase time step (δt_{bub}) and the collision time-step (δt_{coll}) such that:

$$\delta t_{flow} = \mathcal{N}_i \delta t_{bub}, \quad \delta t_{bub} = \sum \delta t_{coll} \tag{8}$$

where $\mathcal{N}_i$ is fixed for a given simulation such that the time integration of all interaction forces acting on a given bubble is carried out in an accurate manner. In this work $\mathcal{N}_i = 20$ to maintain sufficient accuracy for both the force integration and also the DSMC collisions. δt_{coll} is calculated on the fly by the DSMC algorithm from the estimated collision frequency for each discrete phase entity, as reported by Kamath et al. [28]. This is explained in more detail in the coming sections. In DBM, δt_{coll} is calculated as the time-step to reach the next physical collision, with which the whole system is updated.

3.2. Coupling

DSMC algorithms have come far in terms of describing the hydrodynamics and heat transfer, when implemented at the molecular level or in applications where particle-particle or droplet-droplet collisions are the main physical phenomena occurring in the system of interest [32,35,36]. The continuous phase is either absent, weakly coupled or coupled only using momentum based source terms, which results in a completely collision driven system. This is not what we encounter in the current system of bubbly flows. Apart from the right collision frequency, the phase fraction needs to be evaluated for the discretized volume-averaged Navier-Stokes equations (see equation (3) and (4)).

To account for the inter-phase coupling, the gas-fraction is mapped as well as the computation of the volumetric momentum exchange rates between the Eulerian and Lagrangian phases. This is implemented along the lines of Darmana et al. [37] using a polynomial filter function proposed by Deen et al. [38].

Naturally, one would expect an effect of changing the filter width on the predicted time-averaged axial velocity profiles. Fortunately, this was found to be negligible by Lau et al. [39] when a correction to the drag force based on the local void fraction is employed. The filter width is responsible for maintaining ε_l and Φ when a bubble crosses different node points in the Eulerian grid.

Two things have to be monitored and applied with caution in the coupling of the DSMC and the CFD: a) The first is the correction factor of the drag force due to swarm effects as these are defined well for low and intermediate void fractions but not for the dense limit. b) The second is the phase fraction, particularly in the dense regions of the flow, since large overlaps will lead to a singularity in the volume averaged Navier-Stokes equations. To counter the over-packing, additional measures have been taken in the collision algorithm.

3.3. Discrete phase dynamics and collision algorithm

The bubble phase equation of motion is solved using explicit schemes for equation (1):

$$\int_{\mathbf{v}^n}^{\mathbf{v}^{n+1}} d\mathbf{v} = \int_{t^n}^{t^n+\delta t_{bub}} \frac{\sum \mathbf{F}^n}{m_b} dt \tag{9}$$

The force integration has been tested with a fourth order explicit Runge-Kutta scheme and a first order explicit Euler scheme. The first order scheme is more computationally efficient, when the time-step constraint is due to the collisions. Both schemes were checked on a single bubble rise problem (see section 5.2) for many time-steps until the terminal rise velocity is established.

The bubbles are moved in time using the time-marching technique described in section 3.1. During each bubble time step, the forces (using equation (9)) are calculated at the discrete bubble locations using the quantities mapped from the Eulerian grid cells using the polynomial filter described in section 3.2. Due to Newton's third law of motion, a reaction force is collected to be mapped on to the momentum nodes in the Eulerian framework, to be applied at the end of the bubble time-step. The collision sequence is then initiated according to the Algorithm 3.1. This is repeated until the discrete phase has moved for a full δt_{flow}. The forces collected in the volumetric source term are time-averaged over the flow time-step δt_{flow}, since they are calculated multiple times in one flow time-step. The weights for the averaging are calculated based on the ratio $\delta t_{bub}/\delta t_{flow}$ to achieve the correct momentum balance. With the new bubble positions, the volume fraction in each Eulerian cell is calculated using the same polynomial mapping function.

In any particle based technique, the two major steps are the force calculation step and the collision detection step. In a hard-sphere collision model, among which the DSMC, the force calculation and the collision detection loops are separated. Typically the most expensive step is the collision detection step, especially when done trivially leading to a time scaling of the order of $\mathcal{O}(N^2)$ where N is the total number of discrete phase entities in the system. The application of cell lists and neighbour lists reduce the number of comparisons considerably and reduce the time scaling to $\mathcal{O}(pN)$ where the factor p depends on the size of the searching scope or the cut-off distance (r_{cut}). Even then, populating these data structures takes considerable execution time compared to the remaining part of the algorithm. A Verlet list further reduces the number of times the above mentioned data structures need to be populated or refreshed depending on the shell distance R_{shell}. This has been implemented and is being utilized in this work for evaluating the neighbours in the searching scopes of different bubbles.

The modified DSMC collision detection step is detailed in algorithm 3.1. The main additions relative to the already existing algorithm proposed by Pawar et al. [33] are: a) an explicit treatment of nearest neighbour collisions to account for correlated and uncorrelated discrete phase velocities, and b) incorporation of the radial distribution function g_{ij} to account for the increase in collision frequency in dense or clustered arrangements of particles in 3D space [40]. A more detailed explanation can be found in the work of Kamath et al. [28]. A speed up in the DSMC algorithm over the deterministic DBM from Darmana et al. [25] has been reported by Kamath et al. [28] with minimal loss of accuracy for different collision regimes. Moreover, the reason for the speed-up is stated clearly and the serial computational times for a DEM time-step for varied number of bubbles and void fractions have also been reported. In this work, an MPI parallelized version of the same algorithm is presented.

The term g_{ij} is given by Ma and Ahmadi [41] for a mono-disperse system (see equation (10)). This is extended to poly-disperse systems by Santos et al. [42] (see equation (11)).

$$g_{ii} = \frac{(1 + 2.5\varepsilon_p + 4.5904\varepsilon_p^2 + 4.515439\varepsilon_p^3)}{\left(1 - \left(\frac{\varepsilon_p}{\varepsilon_{max}}\right)^3\right)^{0.67802}} \tag{10}$$

$$g_{ij} = \frac{1}{1-\varepsilon_p} + \left[g(r)_{contact,ii} + \frac{1}{1-\varepsilon_p} \right] \frac{d_i d_j}{\bar{d}d_{ij}} \tag{11}$$

Algorithm 3.1 DSMC algorithm.

for each particle id i **do**
  Choose a bubble/particle id $i \in X$ such that $X = \{n : n$ is a positive integer and $n \in N_{tot}\}$ where N_{tot} represents total number of particles in the domain/sub-domain.
  while $t < \delta t_{bub}$ **do**
    Calculate the collision frequency f_i for i based on equation

$$f_i = \sum_{j \in R_{s,i}} |\mathbf{v}_{ij}| \frac{\pi}{4} (d_i^2 + d_j^2) \frac{n_j}{\frac{4}{3}\pi R_{s,i}^3} g_{ij}$$

    where $\mathbf{v}_{ij}$ is the relative velocity between particles/bubbles i and j, d is the diameter, n_j is the parcel size, $R_{s,i}$ is the searching scope size for particle/bubble i and g_{ij} is the radial distribution function at contact for discrete entity i with particle type j.
    $\lambda_i = \frac{|\mathbf{v_i}|}{f_i}$
    $\Delta t_{p,i} = \min\left[\frac{\lambda_i}{3|\mathbf{v_i}|}, \delta t_{bub} - \Delta t_{compl}\right]$
    where $\Delta t_{p,i} \rightarrow$ discrete phase time step, $\lambda_i \rightarrow$ mean free path, $\Delta t_{compl} \rightarrow$ completed time within the time frame of δt_{bub}
    $R_{s,i}^{new} = \max(|\mathbf{v}_i|\Delta t_{p,i}, |\mathbf{v}_{ij}|_{max}\Delta t_{p,i})$
    Choose a random $\chi = \{x \in \mathbb{R} \mid 0 \leq x \leq 1\}$
    $j = int[\chi N_i] + 1$ where $N_i \rightarrow$ number of neighbours in $R_{s,i}$
    $P_{ij} = |\mathbf{v}_i - \mathbf{v}_j| \frac{\pi}{4} (d_i^2 + d_j^2) \frac{n_j \Delta t_{p,i}}{\frac{4}{3}\pi R_{s,i}^3} g_{ij}$
    if $\chi > \frac{j}{N_i} - P_{ij}$ & $(\mathbf{v}_i - \mathbf{v}_j) \cdot (\mathbf{r}_i - \mathbf{r}_j) < 0$ **then**
      Collide discrete phase entities i and j
    else if $Stk_{bub} = \frac{\tau_{bub}}{\tau_l} < 1$ **then**
      Check for nearest neighbour collision
    else
      no collisions
    end if
    Update $\mathbf{r}_i$ and $d\mathbf{r}_i$
    if $|d\mathbf{r}_i| > R_{shell}$ **then**
      neighbour list rebuild = true
    else
      neighbour list rebuild = false
    end if
    $t = t + \Delta t_{p,i}$
  end while
end for

where $\bar{d}$ represents the Sauter mean diameter of the particles within the searching scope, and ε_p is the average solids/bubble volume fraction in the neighbourhood of particle i.

3.3.1. Fluid phase numerical scheme

All numerical simulations from the present work are performed using *FoxBerry* - an in-house developed framework for solving unified transport equations. The discretization of the governing equations (3) and (4) is done on a structured grid with staggered arrangement of variables[2] using the Finite Volume method. The SIMPLE algorithm is used for the pressure-velocity coupling, as described by Patankar [43]. All convective fluxes are discretized with the second order Barton scheme [44] and all diffusive fluxes are discretized with the second order central difference scheme. To keep the numerical stencil compact, a deferred correction method is applied to convective fluxes, resulting in a 7-diagonal matrix in a 3D case. The time integration is carried out using the first order backward Euler method. All discretized terms are treated implicitly, except source terms originating from the discrete phase, which are considered explicitly.

All linear systems are solved by means of an in-house BiCGStab(2) method [45] and the solution of the Pressure Poisson Equation (PPE) is additionally accelerated with the multilevel ML solver [46] from the Trilinos library [47].

4. Parallelization strategy

4.1. Domain decomposition

To obtain the optimal parallel performance, the developed code utilizes a full 3D decomposition instead of the often used pencil or stripe decompositions. This allows for reduction of the sub-domain interface $\mathcal{A}$[3] proportionally to its volume $\mathcal{V}$, which helps to decrease the communication message size between sub-domains and, thus, the communication time.

[2] Discretized scalar fields are stored at the center of a control volume, while components of vector fields are stored at face centers.

[3] To simplify further explanation and discussion, the internal sub-domain cells are referred to as the computational volume $\mathcal{V}$, whereas the cells adjacent to the neighbour sub-domain are called the communication surface area $\mathcal{A}$.

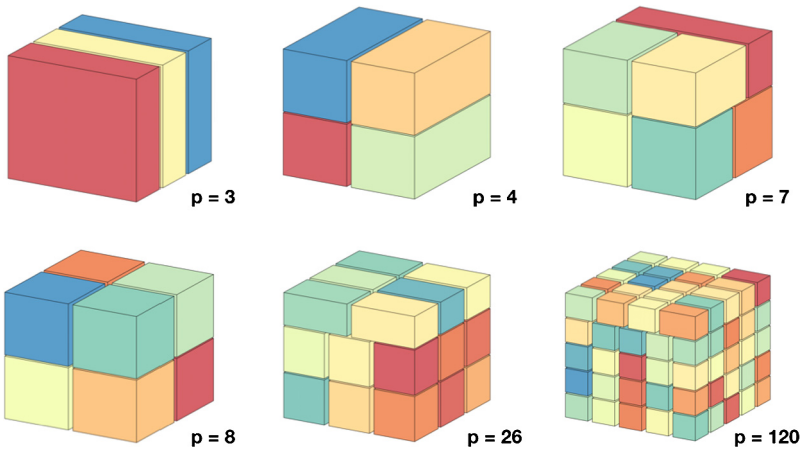


Fig. 1. An example of a full 3D decomposition for various number of processes (p). Sub-domains are coloured randomly for better visualization.

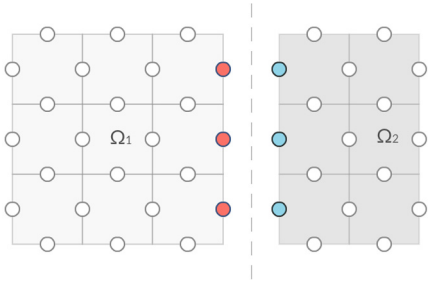


Fig. 2. Distribution of on-edge momentum cells between adjacent processes Ω_1 and Ω_2. Red points represent a copy of blue points which belong to the Ω_2 process.

An example of the cubic domain decomposed with different number of processes is shown in Fig. 1. The decomposition algorithm (Algorithm A.1) is shown in Appendix A.

4.2. Parallelization strategy fluid solver

Due to the staggered arrangement of variables, two adjacent sub-domains Ω_1 and Ω_2 share one layer of momentum cells (see Fig. 2), which results in the duplication of the matrix rows during the assembly step of the momentum matrix. To preserve the initial size of the distributed matrix and to reduce the potential increase of the solution time and the number of iterations of linear solvers, the rule of the "positive owner" is applied. Thus, all shared momentum points on the positive (right) side of the Ω_1 sub-domain are considered as an extra layer of halo cells, while their physical representation belongs to the Ω_2 sub-domain.

The application of the "positive owner" rule results in an extra communication step during the assembly of the Pressure Poisson Equation (PPE), caused by the direct dependence of coefficients in the PPE matrix on the central coefficients of the momentum matrices [43].

To simplify the access to this data, every sub-domain performs a search in its neighbouring sub-domains to determine the ownership of the required elements. This operation is performed once and its result is stored in the form of a hash table, which maps the locally missing and remotely detected elements with the information on corresponding remote process ID. The constructed hash table is used at every iteration of the SIMPLE algorithm to obtain the off-process data and to populate the local vectors of the diagonal elements from the momentum matrices.

4.3. Parallelization strategy DSMC

Parallelization strategy for the DSMC part follows the same domain decomposition technique as the fluid solver mentioned above. Generally particle based simulations follow their own domain decomposition as described in Abraham et al. [48]. Such a decomposition can be uniform or non-uniform, based on the dynamics of the problem itself. Darmana et al. [25] have reported a mirror domain technique to decompose the domain. The advantage of such a method lies in its load balancing, due to an even distribution of bubbles across all domains. The parallel efficiency of such a technique lasts until a maximum within 50 processors, as shown in the same work [25]. As the system becomes larger, the amount of com-

Algorithm 4.1 Bubble phase calculation steps in a sub-domain $\mathcal{P}$.

for $n < \frac{\delta t_{flow}}{\delta t_{bub}}$ **do**
  Set $t_{bub} = 0$.
  Calculate interface forces and velocity of bubbles $\forall\ i \in N_{tot}$
  Bubble/particle removal from the sub-domains/domain
  Bubble/particle transport
  Bubble/particle introduction into the sub-domains/domain
  if neighbour list rebuild = true **then** ▷ see operation 19 in Algorithm 3.1
    Refresh cell-linked lists and neighbour lists $\forall\ i \in N_{tot}$
  else
    Do not update cell linked lists and neighbour lists
  end if
  Transfer real bubble data to neighbouring halo cells
  Execute Algorithm 3.1 $\forall\ i \in N_{tot} + N_{ghost}$
  Calculate volume of bubbles in different cells and map them to the Eulerian sub-domain.
  Apply data reduction to halo cells of current sub-domain to real cells of neighbouring sub-domain for the bubble volumes and the momentum source terms.
end for

munications for such a technique will get larger due to timely synchronizations among processes and as such the parallel efficiency drops quite quickly.

Another important disadvantage of the mirror domain technique is the memory requirement. This is especially apparent when the problem sizes go up such that there are $10^6 - 10^7$ bubbles in the system or even more. Every process should hold enough memory to maintain at least 7 doubles for position and velocity vectors and bubble size, as well as memory for the complex data structures which include the cell linked lists and neighbour lists.

4.4. Algorithm

The overall discrete phase algorithm is described in a combination of Algorithms 4.1 and 3.1. The DSMC algorithm uses the domain decomposition of the Eulerian part itself to avoid extra overhead due to communication of the Eulerian data required for the force calculation. Moreover the two communication steps that are mentioned in Algorithm 4.1 are only between the neighbouring domain processes. The additional performance gain is also due to the fact that local DSMC processes can run almost independently of each other in the sub-domains, unlike the mirror-domain technique employed for the discrete bubble model (see [25]) where the smallest collision time-step across all domains has to be determined and broadcast to all processors.

4.4.1. Data structures and flagging

Cell linked lists and Verlet lists are very well known for collision detection in several particle based techniques [49]. They are always used to minimize the number of particle-pair comparisons to resolve interactions among the particles themselves. The cell linked list used in the current implementation is stored in 2 arrays: one is a list of head *id*s and the other is a list of linked *id*s. An adjacency list is used to store the Verlet list. Apart from these, a particle *id* $\rightarrow$ *cell id* map is also stored to avoid unnecessary *cell id* calculations. The *cell id*s are then further used to identify the location type (using the cell flags) or the neighbourhood of the particles in question.

The memory allocations for these data structures are kept contiguous to avoid excessive refreshing of the CPU cache. Moreover, the *id* storage in the memory for the neighbour list is done in the order of its distance from the particle under consideration. Quick access to these variables for the distance calculation (during the collision frequency calculation) results in as many cache hits as possible (see Fig. 3). This is necessary because the collision frequency calculation is the most expensive step in Algorithm 3.1. SIMD vectorization operations have also shown to improve performance drastically for large numbers of particles [50]. These will be considered for further performance improvements in our future work.

A map of flags on cells is used to identify the location type of the bubbles/particles and also for the boundary detection (see Fig. 4). This map is necessary for boundary detection (like walls) and halo cells, preventing extra checks for the bubbles/particles in the bulk. This flagging is done once during initialization of the simulation, either based on the boundary conditions from the Eulerian part or based on explicit boundary conditions provided for the Lagrangian part.

4.4.2. Bubbles in the halo region

Since the DSMC process in every sub-domain is independent, it is important to check the consistency of the physics in the form of momentum conservation across sub-domains. Several stochastic methods in literature are known not to conserve momentum at the individual particle level (see [27]) but rather at an overall statistical level. Fig. 3 shows a 2D representation of a single layer particle decomposition across neighbouring sub-domains, including the halo regions in colour.

For the pure DSMC algorithm, the exchange of momentum between real and halo particles/bubbles in the current process need not be exactly the same as between the particles/bubbles of halo and real cells of a neighbouring process. This will only happen if the chosen collision pairs are exactly the same across the neighbouring sub-domains. Since the process of

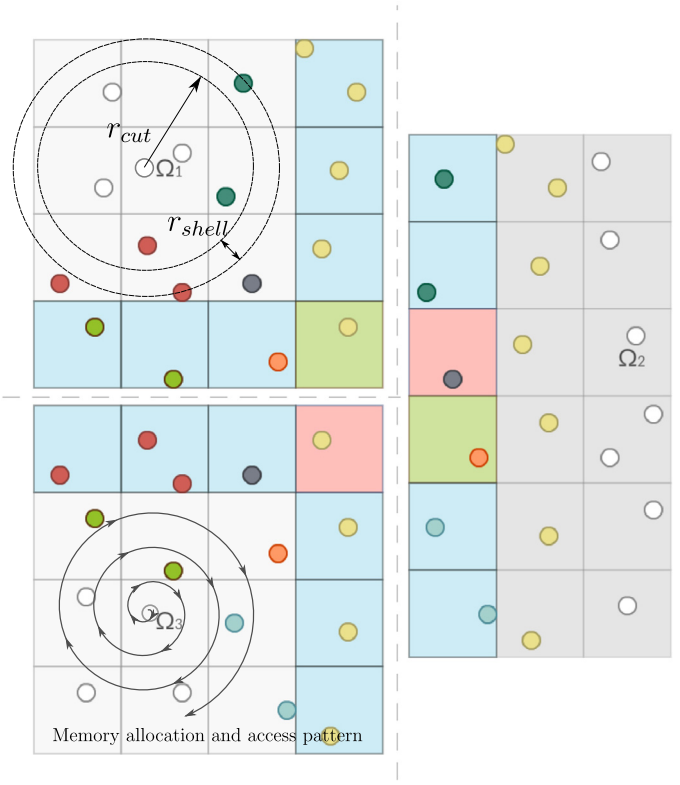


Fig. 3. Decomposition (1 layer of ghost cells) of the discrete phase shown in connected 2D sub-domains with a verlet list illustration at the top and the neighbour-list memory allocation pattern shown at the bottom. Grey cells represent the internal cells and coloured cells represent corresponding halo cells.

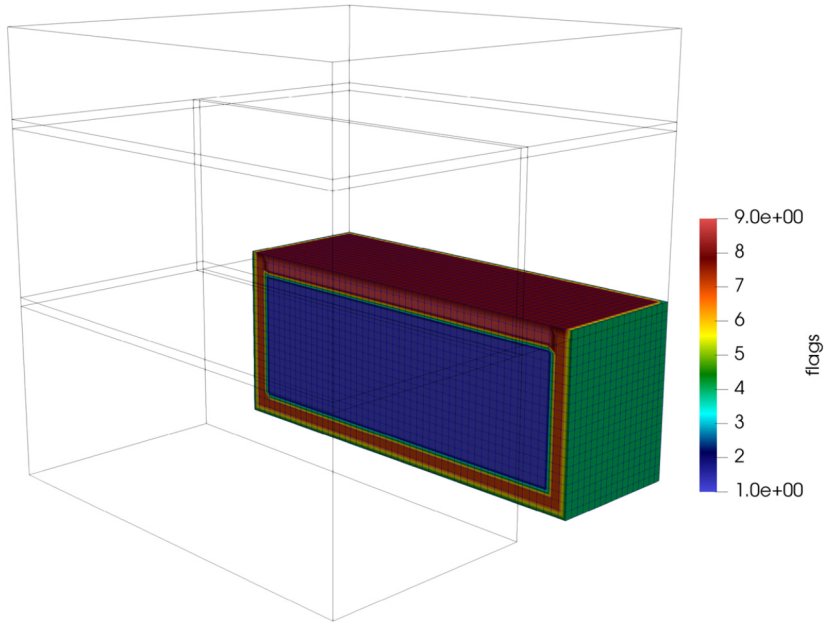


Fig. 4. Map of DSMC flags shown for a sub-domain for a domain decomposition of 5 cores. The flags are numbered based on the type of cells: 1. Internal cell 4. Wall 7. Near a boundary 8. Halo cells 9. Near a process boundary.

collision pair selection is random, the net amount of momentum loss or gain is also random. To minimize the momentum exchange errors, a nearest neighbour collision is used near sub-domain boundaries.

Table 3
Physical properties of the discrete phase and numerical properties of the simulations used in verification tests for the impinging particle stream problem I1.

Parameter	Symbol	I1	I2
Solid density	ρ_l (kg/m^3)	2000	2000
Particle diameter	d_p (mm)	2	2
Particle time step	δt_p (s)	$5 \cdot 10^{-5}$	$5 \cdot 10^{-5}$
Nozzle diameter	D_{nozzle} (m)	0.25	0.1
System size	$L \times W \times H$ (m $\times$ m $\times$ m)	$0.5 \times 0.5 \times 0.5$	$0.25 \times 0.25 \times 0.25$
Inlet mean velocity	v_{mean} (m/s)	2.5	2.5
Standard deviation	v' (m/s)	1.0	1.0

5. Parallel verification and validation

In our previous work [28], we verified and validated the developed DSMC method for two limiting conditions: first for a pure DEM (with no background fluid phase), and second for a fully coupled gas-liquid system. This also included mono-disperse and poly-disperse cases. The calculation of the total number of collisions and also the collision frequency was verified for varying solids and gas fractions. For validation, a comparison was made with the deterministic Discrete Particle Model (DPM) and Discrete Bubble Model (DBM), respectively.

In the current work, similar test cases have been set up to verify the parallel implementation of the DSMC algorithm. Naturally, it cannot be expected from a stochastic algorithm to execute the exact same sequence of collisions when run multiple times, even when run on a single core. This is possible in the deterministic cases when the initial particle distribution is identical every single time. However, for a stochastic algorithm the statistical properties of the collisions and momentum exchange should not change upon parallelization. The following problems are set up for the verification:

- Impinging particle streams
- Single bubble rise across sub-domains
- Bubble columns

5.1. Impinging particle streams

The system consists of two injection nozzles oriented such that they partly face each other. The solid particles flow into the system through these nozzles. The system boundaries act as exits for the particles. The domain is decomposed based on the number of processors provided. The solid particles are provided with inlet velocities based on the overall mass flow rate. To obtain a velocity distribution, a fluctuating velocity, based on a Gaussian distribution is added to the average velocity. The width of the distribution is varied to obtain different velocity profiles from the nozzles. Two test cases, I1 and I2 have been setup (see Table 3 for the simulation parameters) with following goals,

1. To verify with varying number of cores, consistency in the collision frequency and also its independence from the domain decomposition. (I1)
2. To quantify momentum errors that occur across the neighbouring sub-domains because of a mismatch in collision pairs due to independently running DSMC loops. (I2)

A schematic of the problem setup for I1 with different decompositions is reported in Fig. 5. The comparison of the serial version of the method with the deterministic method (DPM) has been reported in Kamath et al. [28] for both mono-disperse and poly-disperse cases at varying solid fractions.

The simulation has been executed for 5 decompositions: 12 cores, 23 cores, 24 cores, 48 cores and 96 cores. The overall collision frequency, the instantaneous collision frequency and the total amount of collisions occurring in the system are reported in Fig. 6. The results are almost independent of the number of cores used with a maximum relative error of 0.2%. Note that this algorithm does not use the nearest neighbour collisions as the $Stk_p \rightarrow \infty$ without any continuous fluid medium.

A smaller representative system of the same problem is simulated (I2) on 5 cores to evaluate the momentum conservation error across neighbouring sub-domains (see Table 3). This error is expected to reduce in time for the DSMC algorithm with respect to the absolute amount of momentum transferred across the sub-domain due to the randomization of the particle *id*s, collision partners and secondly because the searching scope is isotropic.

A schematic representation of the problem along with the process *id* numbering is shown in Fig. 7. The relative errors are calculated for momentum conservation in each sub-domain with all of their neighbours as follows:

$$\Delta_{\text{mom}} = \frac{|\mathbf{p}_a + \mathbf{p}_b|}{|\mathbf{p}_a|} \tag{12}$$

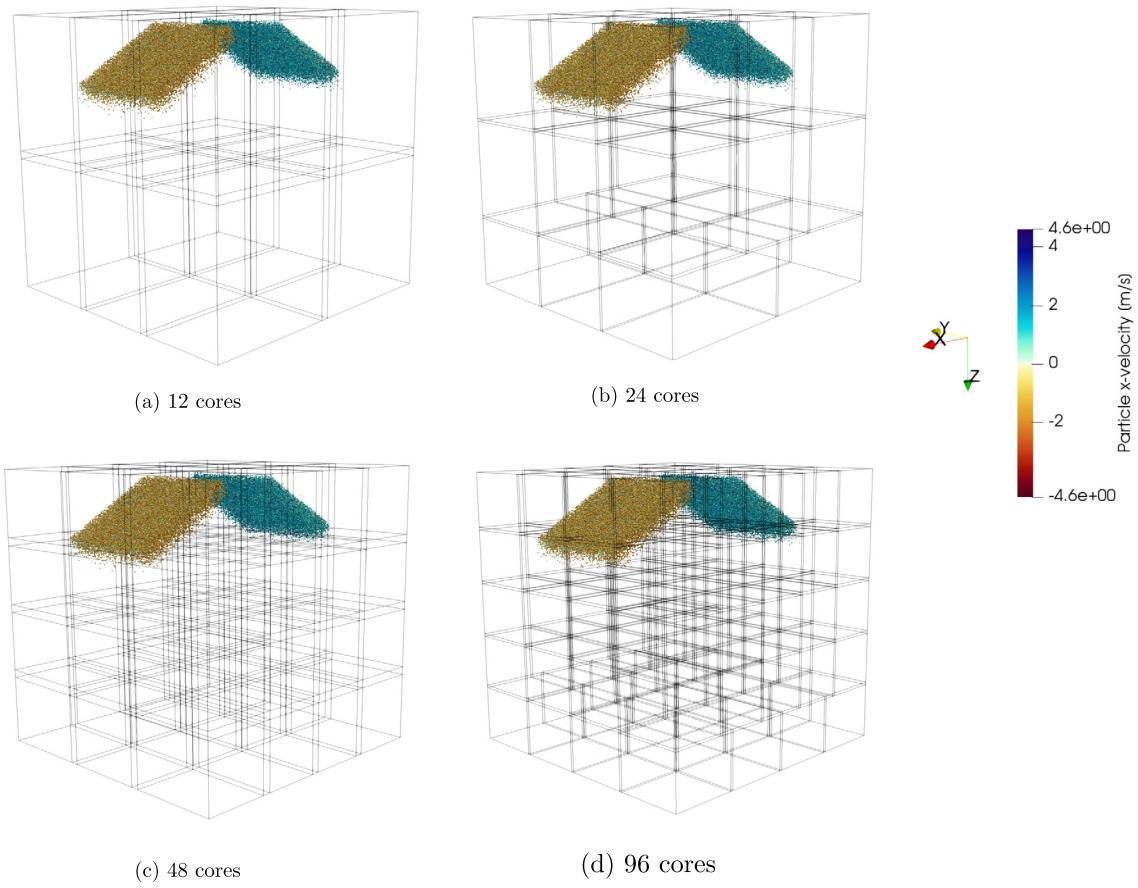


Fig. 5. Impinging particle streams problem setup with different decompositions (I1).

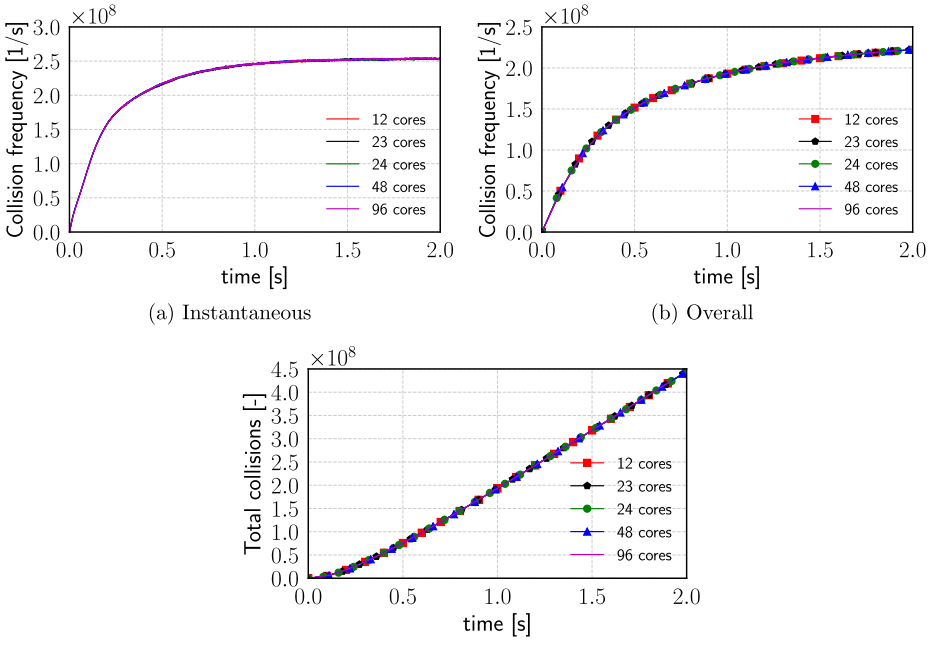
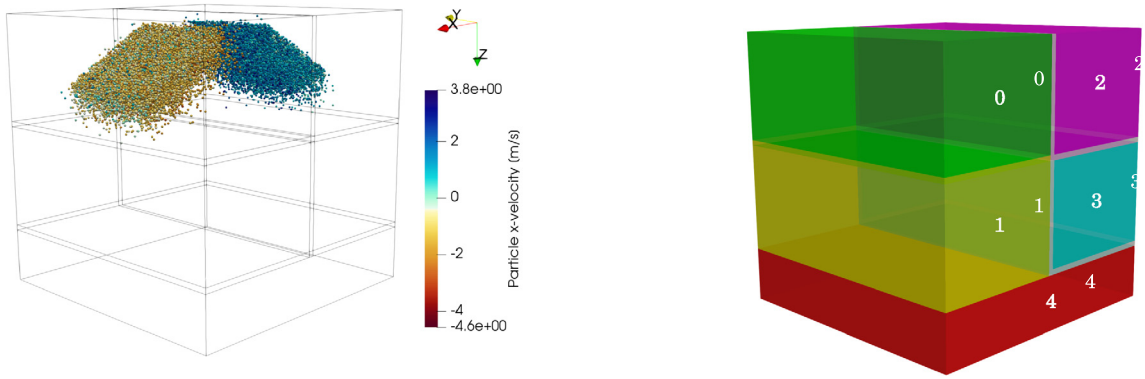


Fig. 6. Collision properties for the impinging particle stream case I1 with the number of cores ranging from 12 to 96.



(a) Particle velocity distribution after 100 particle time steps.

(b) Process id distribution.

Fig. 7. Impinging particle streams problem setup decomposed into 5 processes (I2).

Table 4
Momentum conservation errors across processes due to domain decomposition for the DSMC algorithm reported for the problem I1. Note: This analysis is carried out for a domain decomposition of 5 processes and the errors reported are at the time of 2.0 seconds.

Process boundary	% Error	Process boundary	% Error
$0 \leftrightarrow 1$	1.108	$1 \leftrightarrow 3$	0.331
$0 \leftrightarrow 2$	0.553	$1 \leftrightarrow 4$	1.272
$0 \leftrightarrow 3$	4.921	$2 \leftrightarrow 3$	1.485
$1 \leftrightarrow 2$	5.747	$3 \leftrightarrow 4$	1.204

where $\mathbf{p}_a$ represents the momentum stored in the part of the halo region of sub-process a that is shared with sub-process b and similarly $\mathbf{p}_b$. These momentum errors (Δ_{mom}) are reported in Table 4 and shown for process *id* 0 in Fig. 8. It can be seen that the maximum errors reported are for process *id*s $0 \leftrightarrow 3$ and $1 \leftrightarrow 2$. It should be noted that the overlap between these specific sub-domains is a single row of cells along the diagonal (see Fig. 7b). The net amount of momentum exchanged or the amount of collisions occurring here are nearly two orders of magnitude lower than at the other process boundaries. The cumulative error with respect to the amount of momentum exchanged reduces as more collisions occur.

Nevertheless, to reduce this error, a nearest neighbour collision is forced near the process boundaries in the case of bubbly flows. This reduces the momentum error significantly (to less than 0.5%) as the bubble positions are communicated across the process boundaries and thus the relative distances are the same for a given pair in each sub-process.

5.2. Single bubble rise across sub-domains

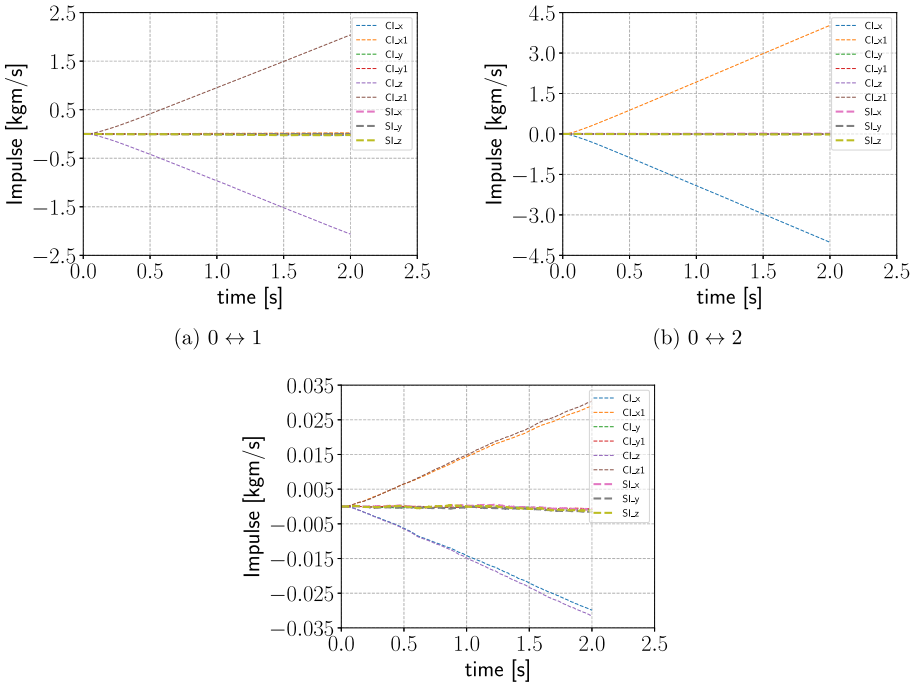
For the second test, a single bubble is initiated in the middle of a domain with six no-slip boundary walls. The test case differs from the previous as the Lagrangian part here is coupled with the flow-solver. Note that because there are no other bubbles to collide with, the purpose of this test is to check whether the domain decomposition has an influence on the momentum coupling between the Lagrangian and Eulerian phase. The simulation settings are specified in Tables 5 and 6. Two tests are carried out to check:

1. the temporal accuracy of the force integration scheme with respect to the chosen time-step.
2. the spatial stability of the force calculation with changing sub-domains.

The temporal accuracy of the first order explicit Euler scheme is checked with a fourth order explicit RK4 scheme (see Fig. 9a). There are negligible differences which indicate the sufficiency of the flow time-step division into different bubble time-steps. It is clearly seen from Fig. 9b that the bubble passes through the sub-domains without any jump in velocity arising due to entry or exit from the sub-domains, which indicates proper communication of flow velocities and the pressure field across sub-domains.

5.3. Bubble columns

Finally, the four-way coupling of the Lagrangian and Eulerian phases is verified via simulation of two tall square cross-sectioned bubble columns (see Fig. 10). Apart from the dimensions, the nozzle distribution in the sparger for both problems is different, ultimately inducing different kinds of flows in both systems. Problem setup parameters are listed in Table 7.



(a) $0 \leftrightarrow 1$ (b) $0 \leftrightarrow 2$

(c) $0 \leftrightarrow 3$

Fig. 8. Cumulative momenta in the three directions collected in the halo regions of the cells belonging to the neighbouring processes of process id 0. Note: CI $\rightarrow$ Cumulative Impulse and SI $\rightarrow$ Sum of the Impulses collected in the halo cells of the respective ids containing common edges ($\mathbf{p}_a + \mathbf{p}_b$).

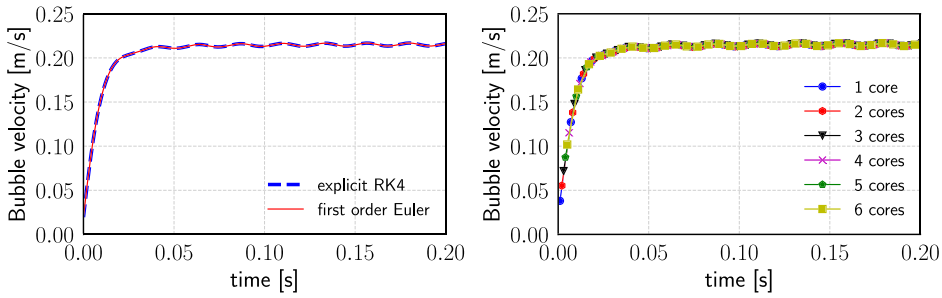
Table 5
Physical properties of the discrete phase and numerical properties of the simulations used in verification tests for the single rising bubble problem.

Parameter	Symbol	Value
Bubble diameter	d_{bub}	4 mm
Bubble time step	δt_{bub}	$5 \cdot 10^{-5}$ s
Box size	$L \times D \times H$	$0.15 \times 0.15 \times 0.15$ (m $\times$ m $\times$ m)
Grid cell size	Δx	5 mm

Table 6
Physical properties of the gas and liquid phases and numerical properties of the simulations used in verification tests.

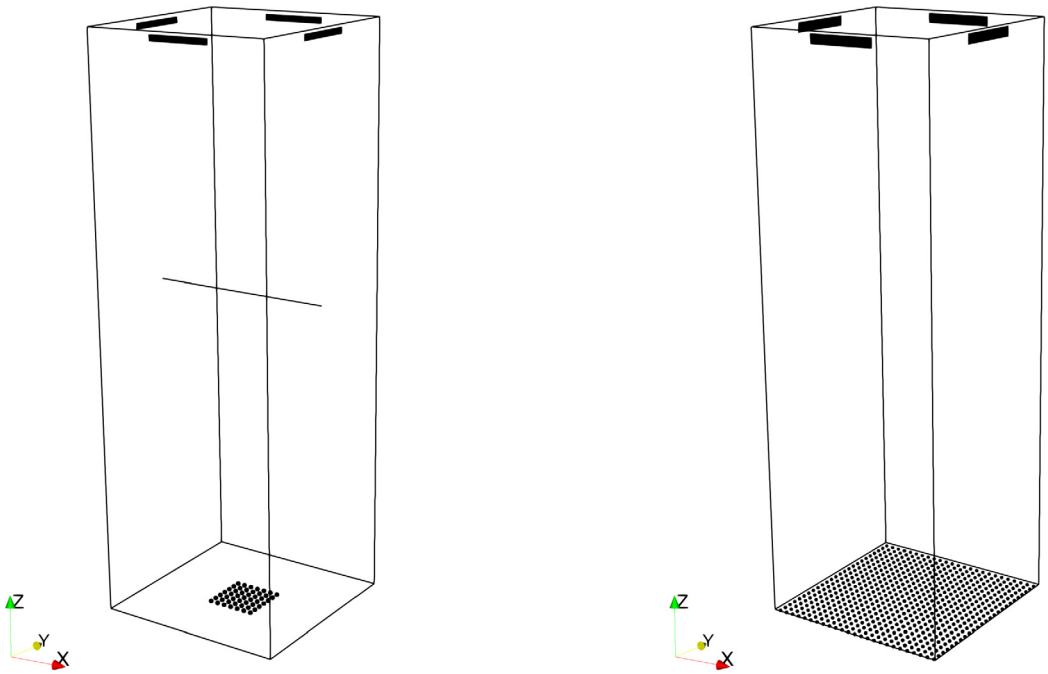
Parameter	Symbol	Value
Liquid density	ρ_l	1000 kg/m^3
Liquid dynamic viscosity	μ_l	$1.002 \cdot 10^{-3}$ kg/m $\cdot$ s
Gas density	ρ_g	1 kg/m^3
Gas dynamic viscosity	μ_g	$1.85 \cdot 10^{-5}$ kg/m $\cdot$ s
Bubble diameter	d_b	4 mm
Surface tension	σ	$72.86 \cdot 10^{-3}$ N/m
Flow time step	δt_f	10^{-3} s
Bubble time step	δt_b	$5 \cdot 10^{-5}$ s
Tolerance	–	10^{-8}

In both cases, the bubbles are introduced into the domain through a number of nozzles arranged in a square cross-sectional arrangement and placed at the bottom of the domain. The superficial gas velocities remain constant during the simulations. On all vertical and bottom walls, a no-slip boundary condition is imposed, whereas the top wall has a free-slip boundary condition mimicking a free surface. To avoid problems with mass conservation when the bubbles are introduced and removed from the system, all vertical walls have an extra slit with a prescribed static pressure outlet boundary condition to facilitate outflow of excess liquid or inflow of liquid into the vacuum created by outflow of bubbles. The slits' locations are symmetrically placed near the top of the columns. The length and height of the slits is equal to one third of the domain's length and one grid cell, respectively.



(a) Varying force integration schemes. (b) Varying domain decomposition.

Fig. 9. Single bubble rise velocity with time.



(a) T1 (b) T2

Fig. 10. Computational domains for the T1 and T2 test cases. Black zones at the top of the columns indicate outlet slits. Black spheres at the bottom represent the sparger. In both cases, the width of the slits is equal to 1/3 of the domain's width, whereas the height of the slits is equal to one and two grid cell sizes for the T1 and T2 cases, respectively. The black line in figure (a) corresponds to a height of $H = 0.25$ m and represents the line along which data is collected.

Table 7
Geometry and sparger properties for the verification tests performed on the square bubble columns. Here, d_p represents the pitch distance, v_b is the superficial gas velocity and D_b is the bubble diameter. Dimensions L, D and H correspond to x, y and z axes, respectively.

Case	$L \times D \times H$, [m]	#nozzles	Pitch d_p, [mm]	D_b, [mm]	v_b, [cm/s]
T1	$0.15 \times 0.15 \times 0.45$	49	6.25	4	0.49
T2	$0.2 \times 0.2 \times 0.6$	625	8	4	1 to 5

The results of the T1 case are compared with time-averaged experimental measurements of Deen et al. [51] and results from the serial code reported earlier [53]. Fig. 11 shows the distribution of the time-averaged axial liquid velocity along the x-axis at a fraction of the column height H of $z = 0.56H$ and fraction of the column depth D of $y = 0.5D$. It can be seen from Fig. 11 that the simulated velocities are in good agreement with the experimental measurements and with the reference data from the serial code. The minor deviations between simulated curves from the present work can be attributed to the stochastic nature of bubble collisions in the DSMC and thus, slightly different behaviour of the plume for each realization. The maxima at approximately $x/L = 0.5$ of the time-averaged axial liquid velocities are slightly higher than

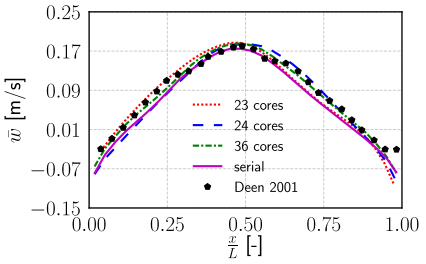


Fig. 11. Time averaged $t = 10 - 200$ s axial liquid velocity profiles. Comparison of execution of the parallel and serial code with experimental measurements of Deen et al. [51].

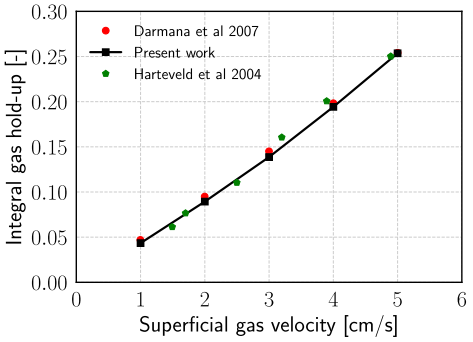


Fig. 12. Dependency of the integral gas hold-up on superficial gas velocity. Comparison of simulated data with reference simulations of Darmana et al. [1] and experimental measurements of Harteveld et al. [52]. Data is averaged over the time interval $10 - 20$ s.

Table 8
Parallel performance of *FoxBerry* code. Comparison of time spent on communication (T_c), solving the momentum equations (T_m), solving the PPE (T_p), matrix assembly (T_a), full time (T_f), the overall speedup and the parallel efficiency. Results are averaged over 10 time steps and 192 cores are taken as a reference.

Cores/192	T_c, [s]	T_m, [s]	T_p, [s]	T_a, [s]	T_f, [s]	Speedup	Efficiency, [%]
1	0.772	21.599	32.030	53.030	116.158	1.0	100.0
2	0.522	12.478	17.206	34.079	69.680	1.7	83.4
4	0.686	6.403	9.963	18.698	37.783	3.1	76.9
8	0.345	4.018	4.928	14.568	27.091	4.3	53.6
16	0.285	2.262	4.248	6.690	15.012	7.7	48.4

those of the serial code. This can be attributed to the nearest-neighbour treatment of bubble-bubble collisions near the edges of the sub-domains. To maintain the momentum conservation across the sub-domain borders which renders it to a slightly more deterministic collision treatment [25].

In the T2 case, results are compared with experimental measurements of Harteveld et al. [52] and numerical data, using the DBM, from Darmana et al. [25]. In this test case, the simulations are performed using 24 cores. Fig. 12 demonstrates the integral gas hold-up as a function of superficial gas velocity. The calculated values of the parallelized code show excellent agreement with the experimental data as well as with the reference simulations.

6. Parallel performance and comparison with previous approaches

In this section, we analyze the parallel performance of the implementation by considering different types of problems. All tests are performed on “Cartesius”, the Dutch National Supercomputer. Every node consists of 2x12 cores of Intel Xeon E5-2690 v3 with a basic clock speed of 2.6 GHz and hyper-threading switched off. The numerical code is compiled using the Intel C++ compiler v16.0.3, Intel MPI v5.1.3 and Trilinos 12.10.

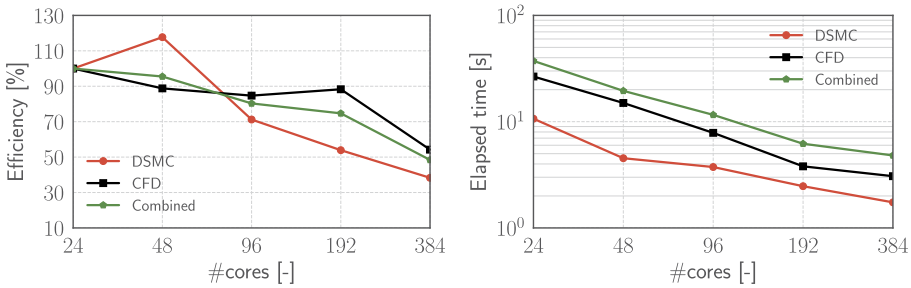
6.1. Single phase flow solver

The parallel efficiency of *FoxBerry* is measured based on the flow in the initial section of the square duct. The domain is discretized on $125 \cdot 10^6$ grid cells. The chosen time step is 10^{-3} s. Each time step takes approximately 20 inner iterations of the SIMPLE algorithm.

The strong scaling of *FoxBerry* is shown in Table 8 and demonstrates 48.4% of the parallel efficiency on 3072 tested cores (relative to the reference performance on 192 cores). The results in Table 8 also show a breakdown of the most

Table 9
Main properties of the P1 and P2 cases. N_{cells} and d_x are the total number of Eulerian cells and their size, respectively. $< N_b >$ is the average number of bubbles in the fully developed system and v_b is the superficial gas velocity. Dimensions L, D and H correspond to x, y and z axes, respectively.

Case	$L \times D \times H$, [m]	N_{cells}	d_x, [mm]	#nozzles
	$< N_b >$	d_p, [mm]	R_b, [mm]	v_b, [cm/s]
P1	$0.45 \times 0.45 \times 1.35$	$2.2 \cdot 10^6$	5	441
	$2.3 \cdot 10^5$	6.25	2	12
P2	$0.50 \times 0.50 \times 2.00$	$4.0 \cdot 10^6$	5	3844
	$0.8 \cdot 10^6$	8.00	2	1



(a) Parallel efficiency. (b) Elapsed time.

Fig. 13. Strong scaling of the P1 case based on 10 iterations. The domain consists of $2.2 \cdot 10^6$ cells and $2.3 \cdot 10^5$ bubbles. The result on 24 cores is taken as a reference.

computationally expensive parts of the code, indicating a significant contribution of the matrix assembly into the total time. This part can be further improved by implementation of a more suitable algorithm for the pressure-velocity coupling for transient problems, e.g. the fractional step method. In addition, the solution of the PPE demonstrates poor scaling after 768 cores, which can be improved with a better preconditioning technique.

6.2. Combined code

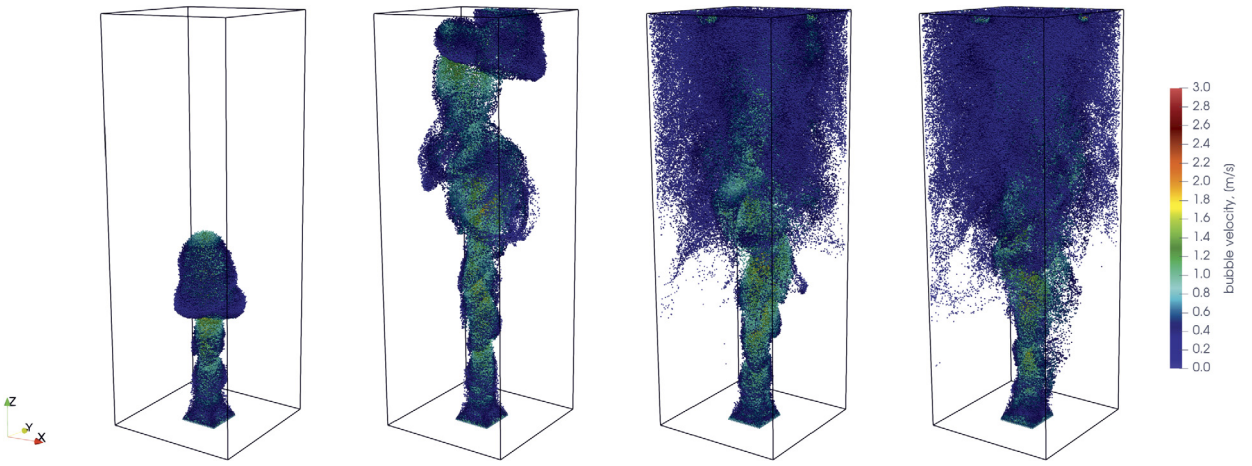
The parallel efficiency of the coupled Euler-Lagrange code directly depends on uniformity of the distribution of Lagrangian entities in the Eulerian domain [4,24]. In bubble column reactors, this distribution is directly determined by the geometry and placement of the sparger. This section demonstrates the parallel efficiency of the developed code on examples of the two most common scenarios where:

(P1) the sparger partially occupies the bottom plate, which leads to appearance of a plume and an imbalanced workload for the parallel DSMC.
(P2) the sparger fully occupies the bottom plate, which leads to uniform bubble rise and a well balanced workload for the parallel DSMC.

The main properties of the aforementioned cases are shown in Table 9. All nozzles are arranged in squares and placed at the bottom of the domain. The governing equations are discretized on a uniform Cartesian grid. All the following results are obtained by averaging the required data over 10 time steps after the flow in the domain is fully developed. It is important to note that the used Barton convection scheme demands 3 layers of halo cells to calculate the convective fluxes coming into a sub-domain. This results in an increase in the communication message size and adversely affects parallel performance when the decomposition is too fine.

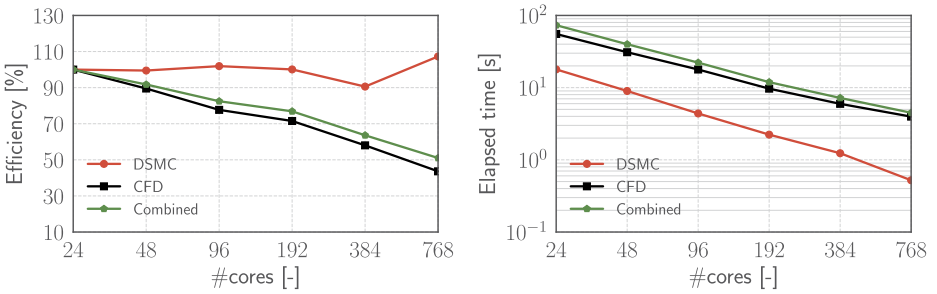
The parallel efficiency and the elapsed time for the P1 case are shown in Fig. 13. The code scales well demonstrating a 48% combined parallel efficiency on 384 cores, each time step takes 4.4 seconds of which 1.6 seconds are taken by the DSMC algorithm. The decrease in the parallel performance of the CFD part is due to the low density of the Eulerian grid, causing MPI communications to be dominant over the workload on each process. Additionally, the non-even distribution of the bubbles (see Fig. 14) results in a significant work unbalance among the processes in the DSMC algorithm. This leads to only 39% parallel efficiency of the DSMC algorithm at 384 cores preventing the coupled code from further scaling.

The problem of unbalanced workload will become more pronounced if the number of bubbles increases, e.g. due to the reduction of their sizes. This issue emphasizes the necessity of having a dynamic domain decomposition for the Lagrangian part in order to improve the parallel performance of the coupled CFD-DSMC code in simulations similar to the P1 case. This requires careful communication of the Eulerian data to the dynamically determined sub-domains for the force calculation



(a) t = 1 s. (b) t = 2 s. (c) t = 10 s. (d) t = 70 s.

Fig. 14. Instantaneous bubble distribution and bubble velocity for the P1 case at different times of the simulation.



(a) Parallel efficiency. (b) Elapsed time.

Fig. 15. Strong scaling of the P2 case based on 10 iterations. The domain consists of $4 \cdot 10^6$ cells and 10^6 bubbles. The result on 24 cores is taken as a reference.

step. This communication step is currently completely absent. Therefore, one also needs to consider if this will actually pay off in terms of performance for general multiphase flow problems.

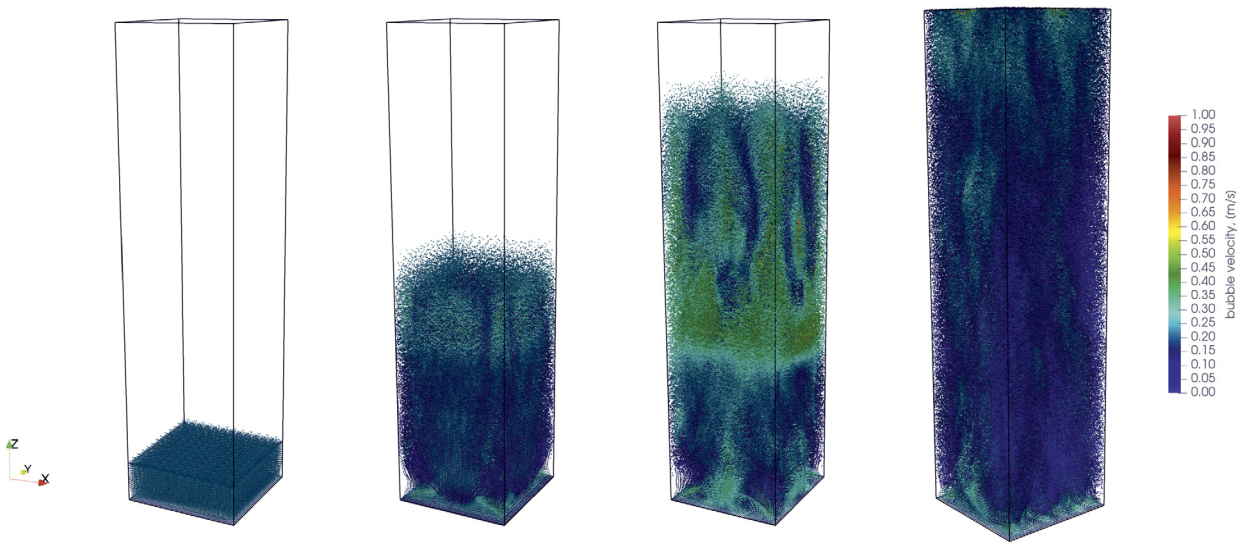
In the P2 case, the absolutely uniform distribution of bubbles in the domain (see Fig. 16) leads to a substantial increase of the parallel performance compared to the P1 case, as shown in Fig. 15. The DSMC algorithm scales up to 768 cores, maintaining excellent parallel efficiency, demonstrating the potential for further scaling. However, the grid density of the Eulerian solver only allows to scale up to 43% parallel efficiency on 768 cores, which leads to an overall parallel performance of 51%. This corresponds to 4.5 seconds elapsed time, among which only 0.5 seconds are taken by the DSMC algorithm.

7. Results

7.1. Large scale bubble column

In this section, a detailed analysis of the hydrodynamics in a large scale bubble column is presented. The configuration of the considered column corresponds to the P1 case described in section 6.2. The simulations are performed for 95 seconds of the physical time and the results are averaged over the time interval between 10 and 95 seconds. The lower boundary of the averaging period corresponds to the time when the fully developed flow conditions are reached.

Fig. 17 shows the distribution of the time-average velocity vector fields at different heights of the column. As can be seen, at the bottom of the column, the liquid velocity reaches approximately 1.3 m/s, while at the top it decreases to almost 0.2 m/s. The high liquid velocity at the bottom is directly related to the high superficial gas velocity and can be explained as follows. The bubbles already released into the domain are decelerated due to the large mass of the liquid above them. Because of a lack of coalescence in this simulation, the new bubbles entering the domain collide with the decelerated bubbles and form a dense swarm. This swarm is pushed by the next sets of bubbles entering the system and thus, it experiences an increase in the axial momentum. At the center of the swarm the liquid fraction is low and bubbles experience less drag. Therefore, they have higher velocities compared to the bubbles at the edges. This dynamics is preserved



(a) t = 1 s. (b) t = 5 s. (c) t = 8 s. (d) t = 50 s.

Fig. 16. Instantaneous bubble distribution and bubble velocity for the P2 case at different times of the simulation.

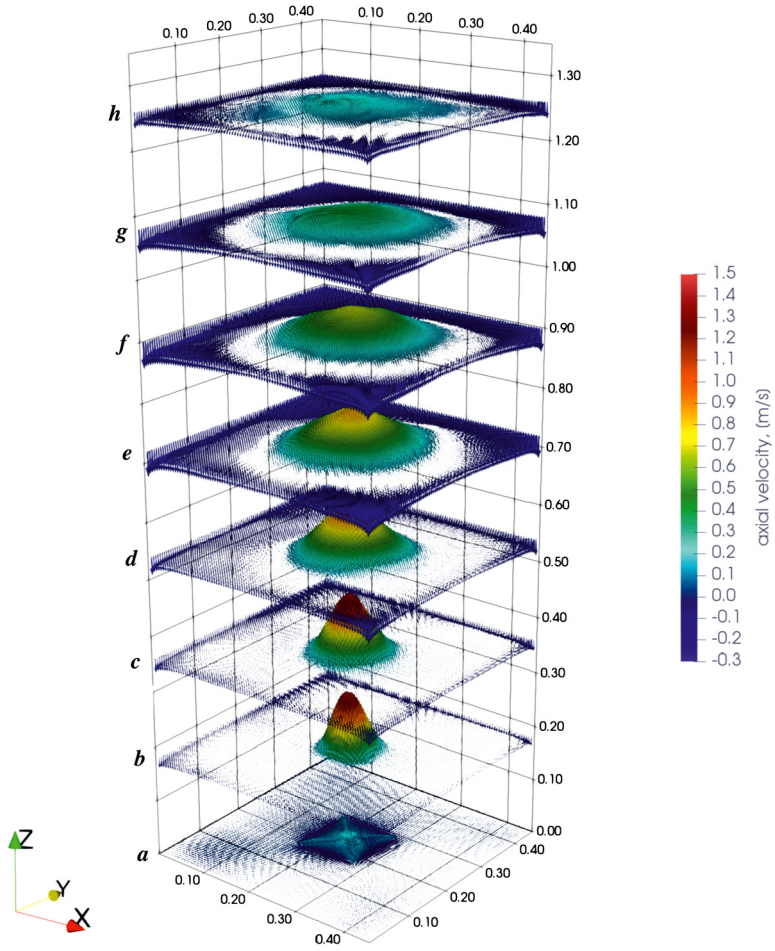


Fig. 17. Distribution of the time averaged velocity vectors at different heights of the domain: a) $H = 0.005$ m; b) $H = 0.18$ m; c) $H = 0.36$ m; d) $H = 0.54$ m; e) $H = 0.72$ m; f) $H = 0.90$ m; g) $H = 1.08$ m; h) $H = 1.26$ m. Data is averaged over the time interval 10...95 s.

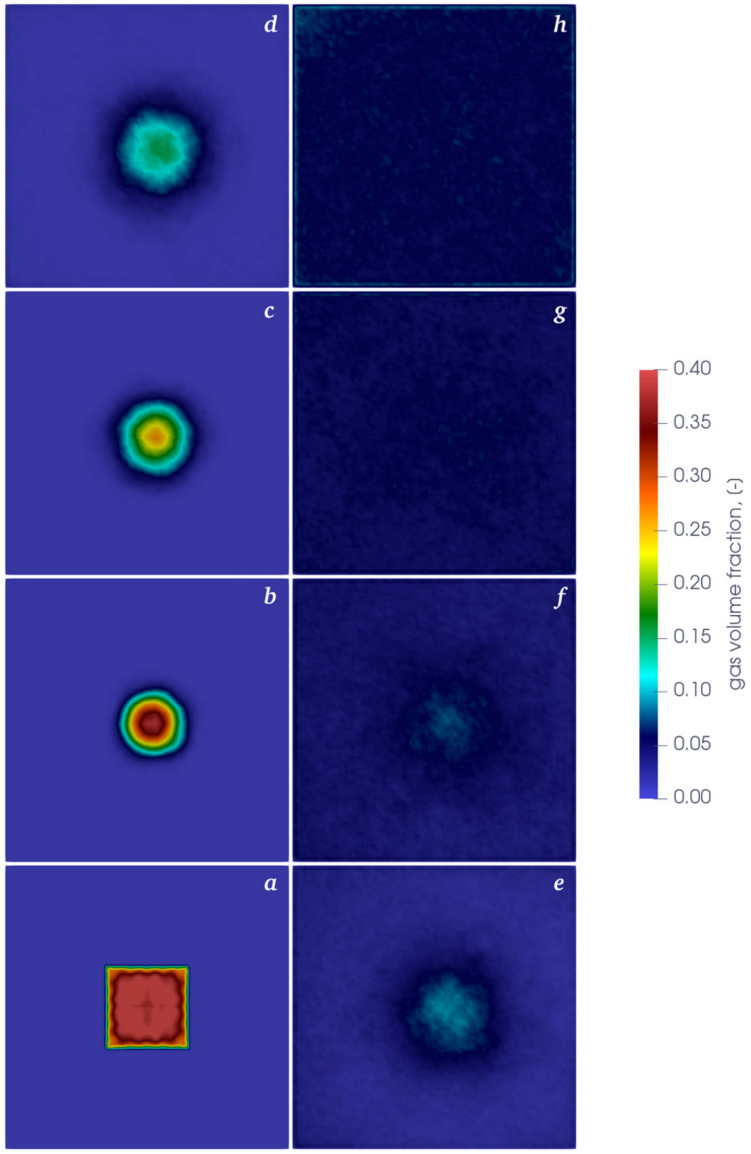


Fig. 18. Distribution of the time averaged gas volume fraction in the xy plane at different heights of the domain. Letters correspond to heights indicated in Fig. 17. Data is averaged over the time interval 10 . . . 95 s.

in time and causes the entrained liquid to behave as a submerged jet which decays along the height due to the viscous forces and presence of the top wall.

Figs. 14c and 14d show that after the flow is developed the overall distribution of the bubbles in the domain remains unchanged. This can be clearly seen from Fig. 18, which shows the time-average gas volume fraction at different heights in the domain. The results show that the column can be clearly split into two parts. The flow in the bottom part is governed by the bubble plume, which maintains the high axial momentum and induces the motion of the liquid. During the simulation, the bubble plume remains compact (see Figs. 18*a-d*) and centered. There is only a slight dispersion along the height. However, the bubble plume breaks at half the height of the domain (see Figs. 18*e-h*). In this part, the bubbles rise at the center of the column while they move down along the side walls, because the force due to the liquid circulation dominates the buoyancy force of the bubbles. At the middle height of the column (see Figs. 17*d-e* and 18*d-e*), the strength of the circulation zones reduces and the downwards moving bubbles are carried away by the more dominant liquid flow at the center of the domain.

The distribution of the time-average vorticity magnitude at different heights of the column is shown in Fig. 19. At the bottom half of the domain the flow is highly rotational around the very pronounced core in the center. The regions of the high vorticity correlate with the swirling effect of the bottom part of the plume, which can be observed in Fig. 14. This swirling effect is caused by the helical trajectories of the bubbles, which is the most energy efficient way for a high-

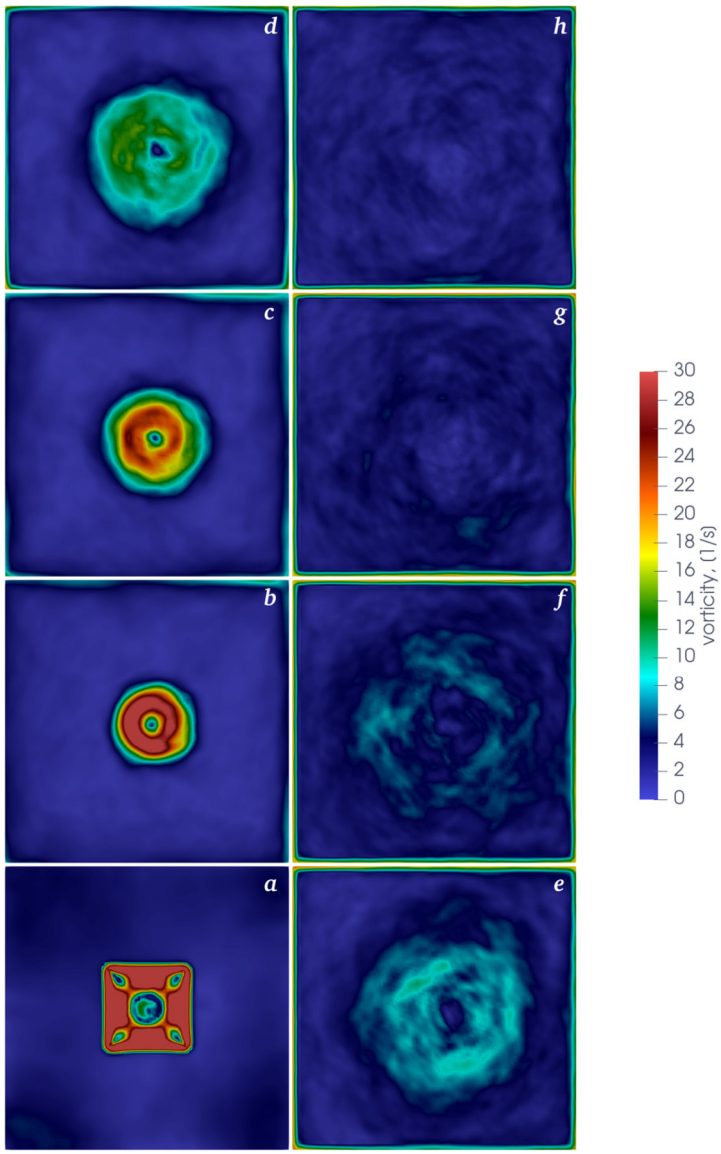


Fig. 19. Distribution of the time averaged vorticity magnitude in the xy plane at different heights of the domain. Letters correspond to heights indicated in Fig. 17. Data is averaged over the time interval 10…95 s. The colour legend is adjusted for better visualization of the low values at the top heights (figures *f-h*). Before the adjustment the maximum value at the bottom (figure *a*) was 50 s^{-1}.

Table 10
The circulation, Γ, at different heights of the column. Γ is calculated by the integration of a vorticity across the full area of the XY cross-sections.

Height	*a*	*b*	*c*	*d*	*e*	*f*	*g*	*h*
Γ, [m^2/s]	0.96	0.55	0.65	0.72	0.90	0.86	0.69	0.59

momentum gas plume to penetrate a viscous fluid. Beside the core, the flow is almost irrotational compared to the central region, except in thin layers near the walls, see Figs. 19*c-d*. At the top half of the column, the plume breakage and the large amount of bubbles lead to the appearance of the rotational motion in the whole domain, which results in the high circulation at $H = 0.72$ m and $H = 0.90$ m, as shown in Table 10. Also, despite the lower magnitude of the vorticity at the top, the circulation in that region is comparable to the one at the bottom.

The aforementioned dynamics can also be seen from the distribution of the turbulence kinetic energy (*TKE*), as shown in Fig. 20. While at the bottom of the column most energetic eddies are concentrated close to the core of the plume, the second half of the column shows a wide region of high *TKE*. Nevertheless, at the top of the column, the *TKE* remains low in the vicinity to the corners and walls, due to viscous effects. In addition, Fig. 20 indicates a low region of the *TKE* at the center

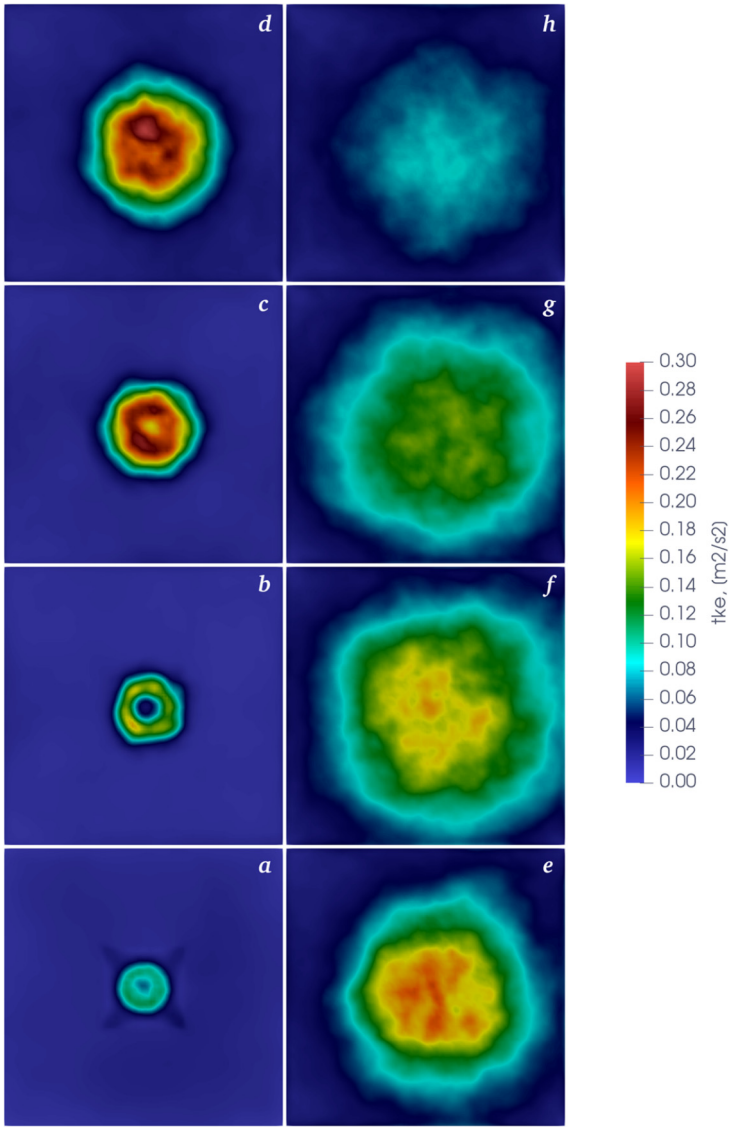


Fig. 20. Distribution of the *TKE* in the xy plane at different heights of the domain. Letters correspond to heights indicated in Fig. 17. Data is averaged over the time interval 10...95 s. The colour legend is adjusted for better visualization of the low values at the top heights (figures *f-h*). Before the adjustment the maximum value at the bottom (figure *a*) was 50 s^{-1}.

of the bubble plume. Similar to a submerged jet, the liquid in this region has a potential core, which can also be observed through the zero vorticity at the center of the bubble plume in Figs. 19*b-c*. However, the *TKE* significantly increases towards the edge of the bubble plume, where the shear-stresses are high and the fluid flow becomes highly turbulent. Further away from the plume, the *TKE* reduces indicating a decrease of the velocity fluctuations.

7.2. Effect of column scale

The results and conclusions obtained from the laboratory-scale bubble column may fail to predict the dynamics in pilot- or industrial-scale columns, which is caused by the influence of the geometry and scale effects. To assess the significance of these effects, three bubble columns of different sizes are compared, see Table 11.

The T1 and P1 cases from Table 11 have already been discussed in section 5.3 and section 7.1, respectively. The P3 case represents a domain, which is extended towards industrial scales from the case P1. The numerical and physical parameters of the P3 case can be found in Table 6. As in all cases, the sparger in the P3 case is represented by nozzles arranged in a square and placed at the center of the bottom of the column. The P3 case is simulated for 27 seconds of the physical time using 768 cores. Both P1 and P3 cases have been simulated for 5 days or a wall time of 120 hours on respective number of physical cores.

Table 11
The comparative list of the main characteristics of laboratory-scale (T1), pilot-scale (intermediate) (P1) and pilot-scale (P3) bubble columns. N_{cells} and d_x are the total number of Eulerian cells and their size, respectively. $< N_b >$ is the average number of bubbles in the fully developed system and v_b is the superficial gas velocity. The dimensions L, D and H correspond to x, y and z axises, respectively.

Case	$L \times D \times H$, [m]	N_{cells}	d_x, [mm]	#nozzles
	$< N_b >$	d_p, [mm]	R_b, [mm]	v_b, [cm/s]
T1	$0.15 \times 0.15 \times 0.45$	$8.1 \cdot 10^4$	5	49
	$5 \cdot 10^3$	6.25	2	0.49
P1	$0.45 \times 0.45 \times 1.35$	$2.2 \cdot 10^6$	5	441
	$2.3 \cdot 10^5$	6.25	2	1
P3	$1 \times 1 \times 10$	10^7	10	625
	$5 \cdot 10^5$	8	4	0.6

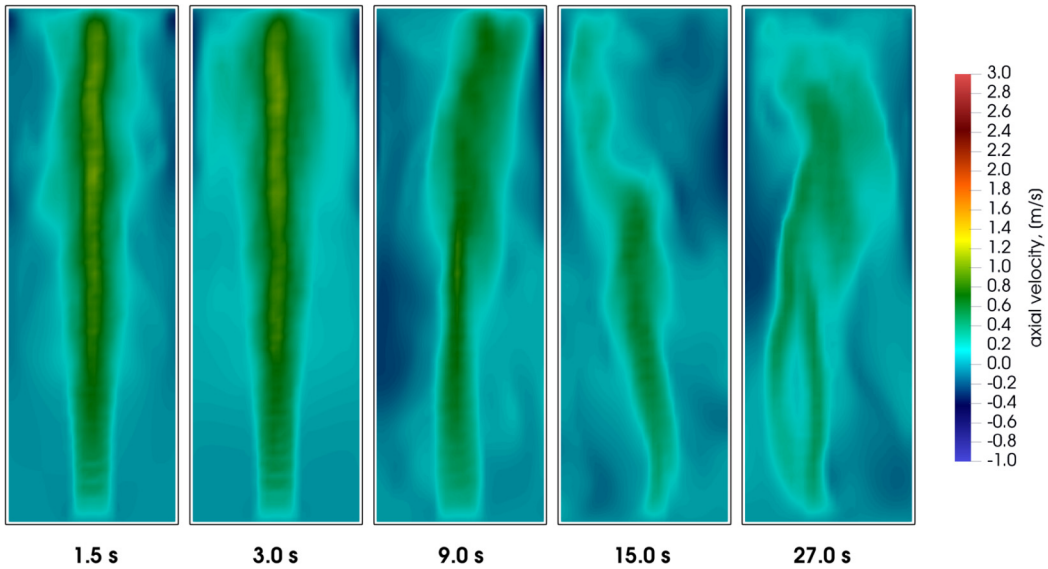


Fig. 21. The instantaneous field of the axial liquid velocity in the mid-plane at different times for the T1 case.

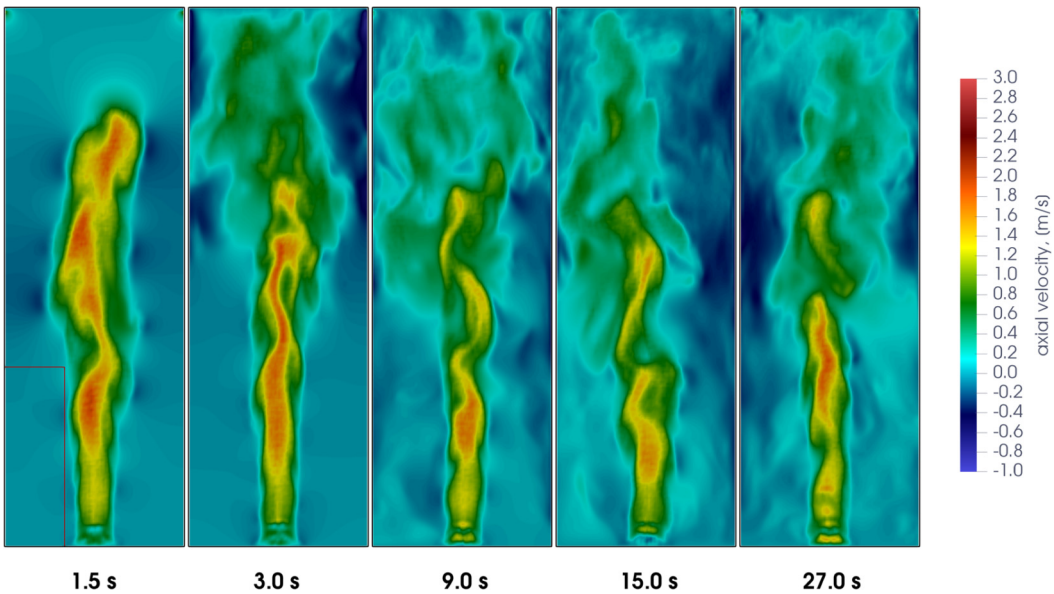


Fig. 22. The instantaneous field of the axial liquid velocity in the mid-plane at different times for the P1 case. The red rectangle indicates the scale of the domain from the T1 case.

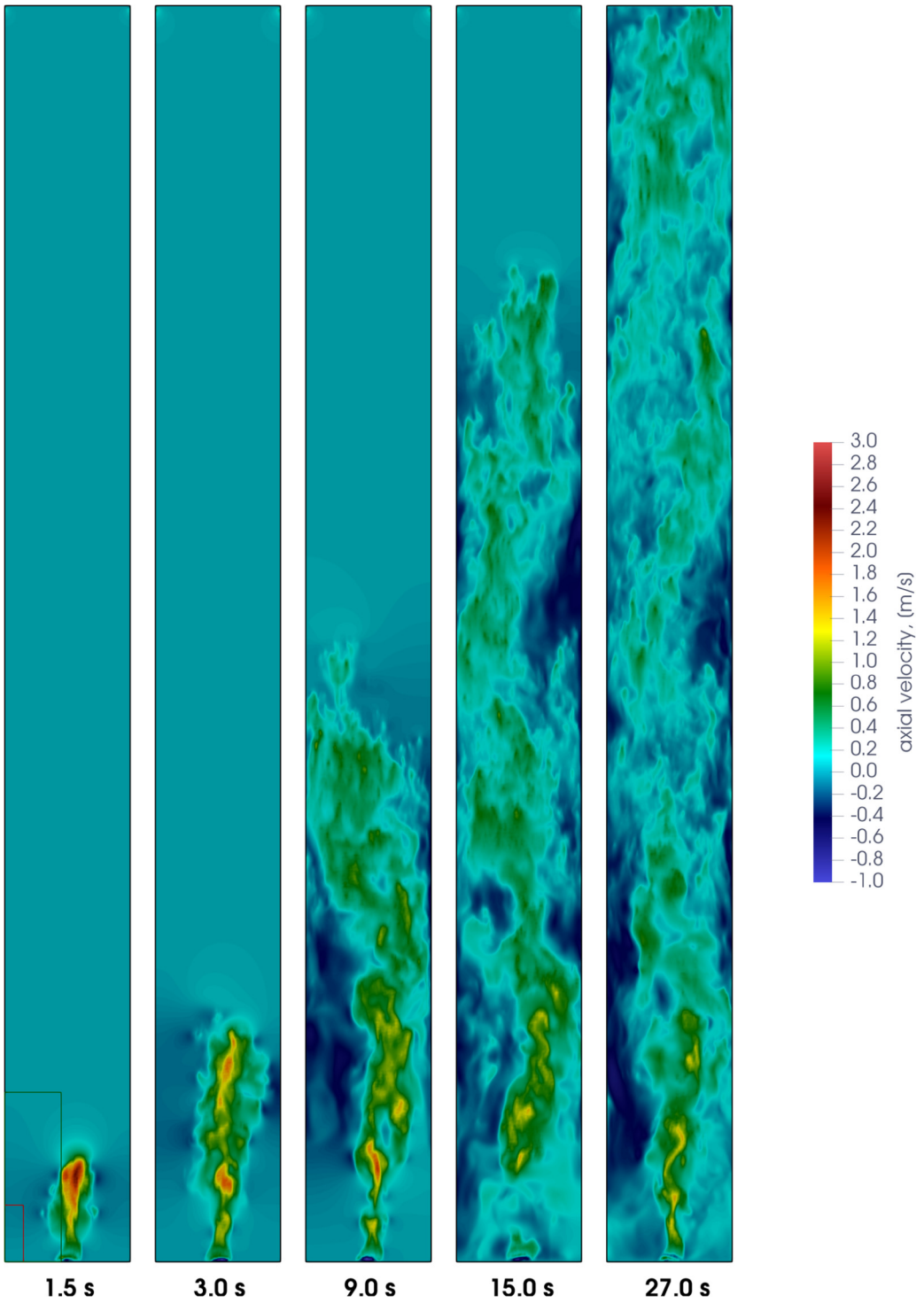


Fig. 23. The instantaneous field of the axial liquid velocity in the mid-plane at different times for the P3 case. The red and green rectangles indicate the scales of the domains from the T1 and P1 cases, respectively.

Figs. 21, 22 and 23 show the instantaneous axial liquid velocity fields at the vertical mid-plane of the domain for the T1, P1 and P3 cases, respectively. It can be clearly seen that the maximal axial liquid velocity in the T1 case varies only slightly along with the height of the column, unlike in the P1 and P3 cases, where the axial liquid velocity is significantly higher at the bottom part of the domain compared to the top part. This is because of the small distance between the opposite vertical walls in the T1 case. Large recirculation zones with a length comparable to the height of the column appear in the

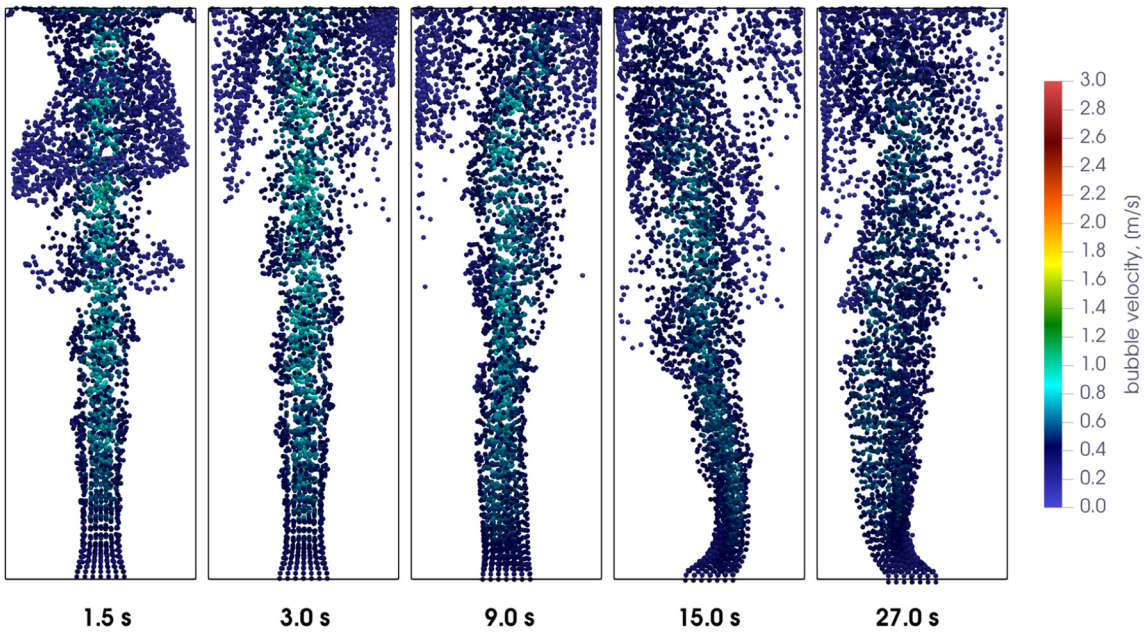


Fig. 24. The instant distribution of the bubbles in the domain at different times for the T1 case.

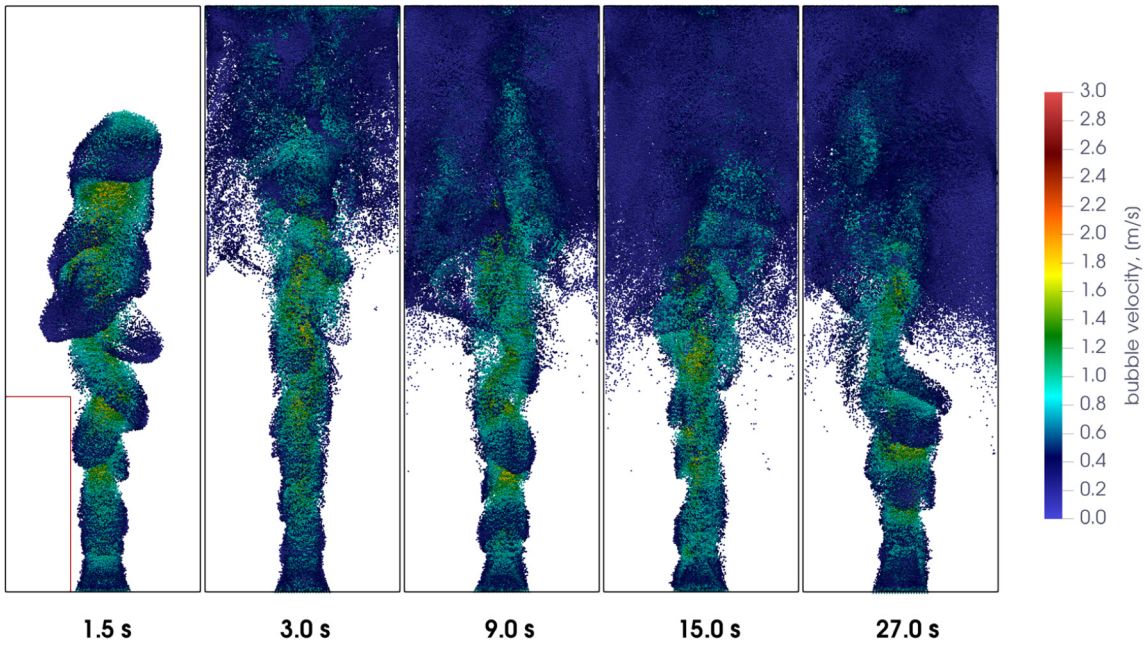


Fig. 25. The instant distribution of the bubbles in the domain at different times for the P2 case. The red rectangle indicates the scale of the domain from the T1 case.

domain shortly after the bubbles reached the top. These circulation zones then travel downwards, disturbing core of the plume leading to meandering of the plume from the very bottom. In contrast, in the P1 and P3 cases, the rise of the bubble plume is accompanied with multiple eddies of different scales from the very beginning. These eddies dissipate energy at different sections of the plume leading to unstable large scale structures that break into smaller ones. The core of the plume is relatively more stable at the bottom because these smaller eddies do not have the energy to disturb the bottom part of the plume completely.

The instantaneous distribution of the bubbles in the three domains is shown in Figs. 24, 25 and 26. The bubble plume in the T1 case is highly influenced by the aforementioned recirculation zones, which push the plume towards the different side walls (see Fig. 24, $t = 15.0$ and 27.0 seconds). Due to the dominance of a single large-scale eddy, the bubble plume may

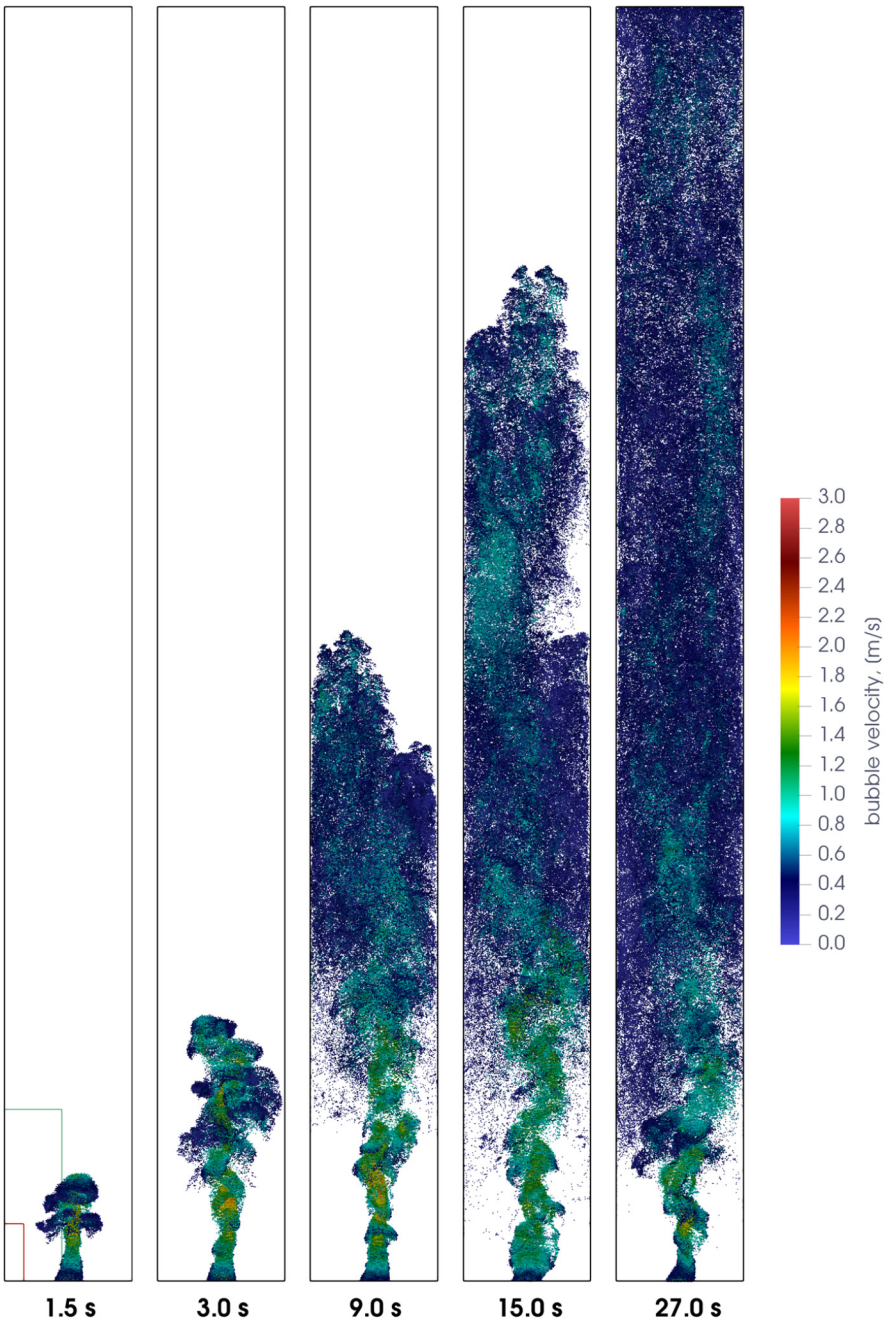


Fig. 26. The instant distribution of the bubbles in the domain at different times for the P3 case. The red and green rectangles indicate the scales of the domains from the T1 and P1 cases, respectively.

stay close to one of the walls for a long period of time, until it is changed by instabilities arising in the bulk. This dynamics is fundamentally different from the P1 and P3 cases, where the plume at the bottom of the domain shifts only slightly from the center and never reaches the walls. In addition, the bubble plume in the P1 and P3 cases breaks shortly after the bubbles enter the domain. Based on the indicated scales in Fig. 26, the breakage of the bubble plume occurs at almost the same heights in both cases. The overall instability of the bubble plume contributes to the almost uniform distribution of the bubbles in the domain above the point of breakage.

The comparison of the bubble distribution in the P1 and P3 cases demonstrates the influence of the height of the domain on the rising bubbles. The front of the rising bubbles in the P3 case consists of constantly appearing small groups of gas inclusions. These groups experience less drag compared to the massive bubble front but they break fast, creating regions with lower pressure. The bubbles from the rising front are attracted by these regions, which leads to the appearance of the

new groups. These dynamics are not observed in the T1 case, where the bubbles rise in a "mushroom-like" manner [1], and in the P1 case, where the rising bubbles represent a dense column moving in the helical path.

8. Conclusion

In this paper a parallel Euler-Lagrange framework and its application to simulation of dense gas-liquid flows are presented. The Lagrangian phase is resolved by means of the DSMC algorithm, for which parallelization is carried out with MPI, whereas the Eulerian phase is resolved using the volume-averaged Navier-Stokes equations incorporated in the in-house numerical framework *FoxBerry*. The coupled parallel code has been verified on several standard problems and demonstrates high accuracy, which is independent of the number of used cores.

The stochastic nature of the DSMC algorithm allows one to avoid limitations of the conventional deterministic (DBM) methods associated with parallelization for distributed memory platforms. Thus, the DSMC method demonstrates a linear scaling up to 768 tested cores for $8.3 \cdot 10^5$ bubbles uniformly distributed across the domain. In the case of non-even distribution of bubbles, the DSMC algorithm allows one to simulate $2.3 \cdot 10^5$ gas entities using 384 cores and maintaining 39% parallel efficiency.

The developed algorithms allow, for the first time, to simulate and analyze a tall square cross-sectioned bubble column using full four-way coupling of the Eulerian and Lagrangian phases. The results demonstrate a pronounced split of the domain in two sections. In the bottom section, the bubble plume occupies only the central part of the domain and determines the overall dynamics of the Eulerian phase. Close to the center of the domain the liquid demonstrates a high rotational motion, which is related to the swirling behaviour of the bubble plume. In the top section the plume breaks, which leads to a more uniform distribution of bubbles in the bulk and the appearance of a large amount of unstable eddies. This splitting is not observed in a smaller scale bubble column, where the bubble plume remains pronounced along the whole height of the domain and the dynamics of the plume is governed by the large-scale turbulent structures.

Declaration of competing interest

The authors declare that they have no known conflicts of interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

This work is part of the TOP grant research programme "First principles based multi-scale modeling of transport in reactive three phase flows" with project number 716.014.001, financed by the Netherlands Organisation for Scientific Research (NWO). Also, this work was supported by the Netherlands Center for Multiscale Catalytic Energy Conversion (MCEC), which is an NWO Gravitation programme funded by the Ministry of Education, Culture and Science of the government of the Netherlands. The authors also thank SURF SARA (www.surfsara.nl) and NWO for the support in using the Cartesius supercomputer.

Appendix A. Domain decomposition

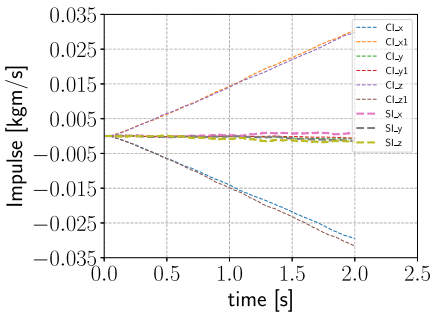
Algorithm A.1 Full 3D domain decomposition.

Input: $\vec{N}$ - number of cells in the domain in each direction, p -number of processes

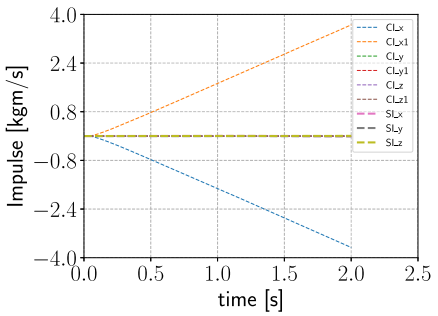
1: $\vec{\xi} = \{\text{round}(\sqrt[3]{p}), \text{round}(p/\xi_1^2), \text{floor}(p/(\xi_1\xi_2))\}$ ▷ Basic number of sub-domains in each direction
2: $\vec{d} = \vec{N} \oslash \vec{\xi}$ ▷ Number of sub-elements in each sub-domain
3: *Perform a full 3D decomposition of the domain using* $\vec{\xi}$ *and* $\vec{d}$
4: $s = \prod_{k=1}^{3} \xi_k$ ▷ Number of decomposed domains
5: $r = |s - p|$ ▷ Remaining sub-domains
6: **if** $r \neq 0$ **then**
7: $\vec{\chi} = \{\text{round}(\sqrt{r}), r/\chi_1\}$ ▷ Remaining number of pseudo-2D sub-domains
8: $\vec{q} = \vec{N} \oslash \vec{\chi}$ ▷ Number of sub-elements in each sub-domain
9: $\eta = \xi_3 d_3$ ▷ Starting index in 3rd direction for pseudo-2D decomposition
10: *Perform pseudo-2D decomposition of the remaining domain using* $\vec{\chi}$ *and* $\vec{q}$*, starting from* η
11: *Build maps between halo and real cells*
12: **end if**

Appendix B. Momentum error all processes

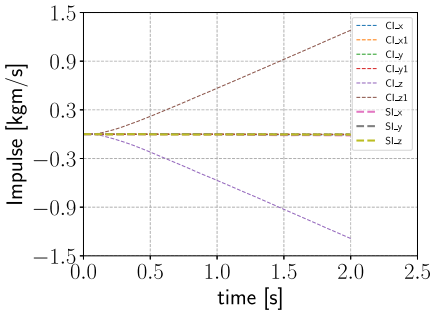
See Fig. B.27.



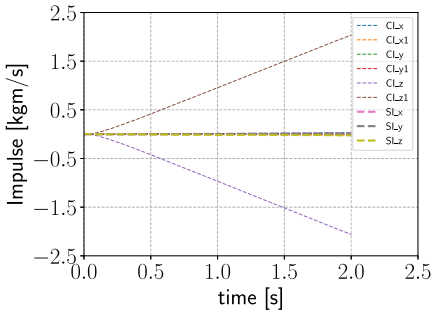
(a) $1 \leftrightarrow 2$



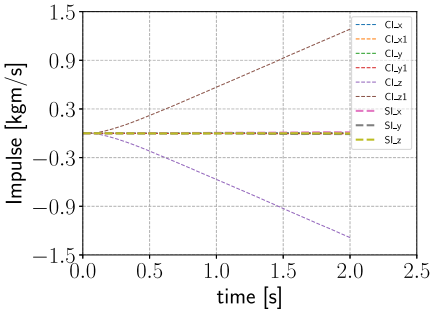
(b) $1 \leftrightarrow 3$



(c) $1 \leftrightarrow 4$



(d) $2 \leftrightarrow 3$



(e) $3 \leftrightarrow 4$

Fig. B.27. Cumulative momenta in the three directions collected in the halo regions of the cells belonging to the neighbouring processes of the remaining process ids. Note: CI → Cumulative Impulse and SI → Sum of the Impulses collected in the halo cells of the respective ids.

References

[1] D. Darmana, N.G. Deen, J.A.M. Kuipers, Parallelization of an Euler-Lagrange model using mixed domain decomposition and a mirror domain technique: application to dispersed gas liquid two-phase flow, J. Comput. Phys. 220 (1) (2006) 216–248, https://doi.org/10.1016/j.jcp.2006.05.011.

[2] P. Ouro, B. Fraga, U. Lopez-Novoa, T. Stoesser, Scalability of an Eulerian-Lagrangian large-eddy simulation solver with hybrid MPI/OpenMP parallelisation, Comput. Fluids 179 (2019) 123–136, https://doi.org/10.1016/j.compfluid.2018.10.013.

[3] A.D. Rakotonirina, A. Wachs, Grains3D, a flexible DEM approach for particles of arbitrary convex shape - Part II: parallel implementation and scalable performance, Powder Technol. 324 (2018) 18–35, https://doi.org/10.1016/j.powtec.2017.10.033.

[4] G. Pozzetti, X. Besseron, A. Rousset, B. Peters, A co-located partitions strategy for parallel CFD–DEM couplings, Adv. Powder Technol. (2018), https://doi.org/10.1016/j.apt.2018.08.025.

[5] Y. Tian, S. Zhang, P. Lin, Q. Yang, G. Yang, L. Yang, Implementing discrete element method for large-scale simulation of particles on multiple GPUs, Comput. Chem. Eng. 104 (2017) 231–240, https://doi.org/10.1016/j.compchemeng.2017.04.019.

[6] V. Spandan, V. Meschini, R. Ostilla-Mónico, D. Lohse, G. Querzoli, M.D. de Tullio, R. Verzicco, A parallel interaction potential approach coupled with the immersed boundary method for fully resolved simulations of deformable interfaces and membranes, J. Comput. Phys. 348 (2017) 567–590, https://doi.org/10.1016/j.jcp.2017.07.036.

[7] S. Wang, K. Luo, S. Yang, C. Hu, J. Fan, Parallel LES-DEM simulation of dense flows in fluidized beds, Appl. Therm. Eng. 111 (2017) 1523–1535, https://doi.org/10.1016/j.applthermaleng.2016.07.161.

[8] N. Geneva, C. Peng, X. Li, L.-P. Wang, A scalable interface-resolved simulation of particle-laden flow using the lattice Boltzmann method, Parallel Comput. 67 (2017) 20–37, https://doi.org/10.1016/j.parco.2017.07.005.
[9] A. Loisy, Direct Numerical Simulation of bubbly flows: coupling with scalar transport and turbulence, Ph.D. thesis, Université de Lyon, 2017.
[10] P. Liu, T. Brown, W.D. Fullmer, T. Hauser, C. Hrenya, R. Grout, H. Sitaraman, A Comprehensive Benchmark Suite for Simulation of Particle Laden Flows Using the Discrete Element Method with Performance Profiles from the Multiphase Flow with Interface eXchanges (MFiX) Code, Tech. Rep. NREL/TP-2C00-65637, National Renewable Energy Laboratory, 2016.
[11] X. Yue, H. Zhang, C. Ke, C. Luo, S. Shu, Y. Tan, C. Feng, A GPU-based discrete element modeling code and its application in die filling, Comput. Fluids 110 (2015) 235–244, https://doi.org/10.1016/j.compfluid.2014.11.020, parCFD 2013.
[12] L. Mountrakis, E. Lorenz, O. Malaspinas, S. Alowayyed, B. Chopard, A.G. Hoekstra, Parallel performance of an IB-LBM suspension simulation framework, J. Comput. Sci. 9 (2015) 45–50, https://doi.org/10.1016/j.jocs.2015.04.006, Computational Science at the Gates of Nature.
[13] M. Jingsen, H. Chao-Tsung, G.L. Chahine, Shared-memory parallelization for two-way coupled Euler–Lagrange modeling of cavitating bubbly flows, J. Fluids Eng. 137 (2004) 207–224.
[14] C. Goniva, B. Blais, S. Radl, C. Kloss, Open source CFD-DEM modelling for particle-based processes, in: 11th International Conference on Computational Fluid Dynamics in the Minerals and Process Industries, 2015, Conference date: 07-12-2015 through 09-12-2015.
[15] R. Berger, C. Kloss, A. Kohlmeyer, S. Pirker, Hybrid parallelization of the LIGGGHTS open-source DEM code, Powder Technol. 278 (2015) 234–247, https://doi.org/10.1016/j.powtec.2015.03.019.
[16] C. Wu, O. Ayeni, A. Berrouk, K. Nandakumar, Parallel algorithms for CFD–DEM modeling of dense particulate flows, Chem. Eng. Sci. 118 (2014) 221–244, https://doi.org/10.1016/j.ces.2014.07.043.
[17] C. Niethammer, S. Becker, M. Bernreuther, M. Buchholz, W. Eckhardt, A. Heinecke, S. Werth, H.-J. Bungartz, C.W. Glass, H. Hasse, J. Vrabec, M. Horsch, ls1 mardyn: the massively parallel molecular dynamics code for large systems, J. Chem. Theory Comput. 10 (10) (2014) 4455–4464, https://doi.org/10.1021/ct500169q, pMID: 26588142.
[18] H. Liu, D.K. Tafti, T. Li, Hybrid parallelism in MFIX CFD-DEM using OpenMP, Powder Technol. 259 (2014) 22–29, https://doi.org/10.1016/j.powtec.2014.03.047.
[19] A. Amritkar, S. Deb, D. Tafti, Efficient parallel CFD-DEM simulations using OpenMP, J. Comput. Phys. 256 (2014) 501–519, https://doi.org/10.1016/j.jcp.2013.09.007.
[20] K. ChihKuang, S. Jaeheon, H. Ezeldin, S. Wei, Parallel Eulerian-Lagrangian method with adaptive mesh refinement for moving boundary computation, J. Fluids Eng. 137 (2004) 207–224.
[21] D. Kafui, S. Johnson, C. Thornton, J. Seville, Parallelization of a Lagrangian–Eulerian dem/cfd code for application to fluidized beds, Powder Technol. 207 (1) (2011) 270–278, https://doi.org/10.1016/j.powtec.2010.11.008.
[22] A. Maknickas, A. Kaceniauskas, R. Kacianauskas, R. Balevicius, A. Dziugys, Parallel dem software for simulation of granular media, Informatica 17 (2006) 207–224.
[23] T. Pohl, F. Deserno, N. Thurey, U. Rude, P. Lammers, G. Wellein, T. Zeiser, Performance evaluation of parallel large-scale lattice Boltzmann applications on three supercomputing architectures, in: SC'04: Proceedings of the 2004 ACM/IEEE Conference on Supercomputing, 2004, p. 21.
[24] G. Pozzettia, H. Jasakb, X. Besserona, A. Rousseta, B. Petersa, A Parallel Dual-Grid Multiscale Approach to CFD-DEM Couplings, 2018.
[25] D. Darmana, On the Multiscale Modelling of Hydrodynamics, Mass Transfer and Chemical Reactions in a Bubble Column, PrintPartners Ipskamp, Enschede, the Netherlands, 2006.
[26] R. Sungkorn, J.J. Derksen, J.G. Khinast, Modeling of turbulent gas–liquid bubbly flows using stochastic Lagrangian model and lattice-Boltzmann scheme, Chem. Eng. Sci. 66 (12) (2011) 2745–2757.
[27] M. Sommerfeld, Validation of a stochastic Lagrangian modelling approach for inter-particle collisions in homogeneous isotropic turbulence, Int. J. Multiph. Flow 27 (10) (2001) 1829–1858.
[28] S. Kamath, J.T. Padding, K.A. Buist, J.A.M. Kuipers, Stochastic DSMC method for dense bubbly flows: methodology, Chem. Eng. Sci. 176 (2017) 454–475.
[29] A. Tomiyama, T. Matsuoka, T. Fukuda, T. Sakaguchi, A simple numerical method for solving an incompressible two-fluid model in a general curvilinear coordinate system, Multiph. Flow 95 (1995).
[30] A. Tomiyama, H. Tamai, I. Zun, S. Hosokawa, Transverse migration of single bubbles in simple shear flows, Chem. Eng. Sci. 57 (11) (2002) 1849–1858.
[31] I. Roghair, Y.M. Lau, N.G. Deen, H.M. Slagter, M.W. Baltussen, M. Van Sint Annaland, J.A.M. Kuipers, On the drag force of bubbles in bubble swarms at intermediate and high Reynolds numbers, Chem. Eng. Sci. 66 (14) (2011) 3204–3211.
[32] G.A. Bird, Molecular gas dynamics and direct simulation of gas flows, https://doi.org/10.1016/0042-207X(96)80021-2, 1994.
[33] S.K. Pawar, J.T. Padding, N.G. Deen, A. Jongsma, F. Innings, J.A.M. Kuipers, Lagrangian modelling of dilute granular flow modified stochastic DSMC versus deterministic DPM, Chem. Eng. Sci. 105 (2014) 132–142, https://doi.org/10.1016/j.ces.2013.11.004.
[34] A.W. Vreman, An eddy-viscosity subgrid-scale model for turbulent shear flow: algebraic theory and applications, Phys. Fluids 16 (10) (2004) 3670–3681.
[35] G. Bird, Sophisticated DSMC, in: DSMC07 Meeting, 2007, pp. 1–49, http://www.gab.com.au/dsmc07notes.pdf.
[36] S. Pawar, J. Padding, N. Deen, A. Jongsma, F. Innings, J.A.M. Kuipers, Numerical and experimental investigation of induced flow and droplet–droplet interactions in a liquid spray, Chem. Eng. Sci. 138 (2015) 17–30, https://doi.org/10.1016/j.ces.2015.07.048.
[37] D. Darmana, R.L.B. Henket, N.G. Deen, J.A.M. Kuipers, Detailed modelling of hydrodynamics, mass transfer and chemical reactions in a bubble column using a discrete bubble model: chemisorption of into NaOH solution, numerical and experimental study, Chem. Eng. Sci. 62 (9) (2007) 2556–2575, https://doi.org/10.1016/j.ces.2007.01.065.
[38] N.G. Deen, M. van Sint Annaland, J.A.M. Kuipers, Multi scale modeling of dispersed gas liquid two-phase flow, Chem. Eng. Sci. 59 (8–9) (2004) 1853–1861, https://doi.org/10.1016/j.ces.2004.01.038.
[39] Y.M. Lau, I. Roghair, N.G. Deen, M. van Sint Annaland, J.A.M. Kuipers, Numerical investigation of the drag closure for bubbles in bubble swarms, Chem. Eng. Sci. 66 (14) (2011) 3309–3316.
[40] S. Sundaram, L.R. Collins, Collision statistics in an isotropic particle-laden turbulent suspension. Part 1. Direct numerical simulations, J. Fluid Mech. 335 (1997) 75–109, https://doi.org/10.1017/S0022112096004454.
[41] D. Ma, G. Ahmadi, An equation of state for dense rigid sphere gases, J. Chem. Phys. 84 (6) (1986) 3449–3450.
[42] A. Santos, S.B. Yuste, M.L. de Haro, Equation of state of a multicomponent d-dimensional hard-sphere fluid, Mol. Phys. 96 (1) (1999) 1–5.
[43] S. Patankar, Numerical Heat Transfer and Fluid Flow, CRC Press, 1980.
[44] J. Centrells, J. Wilson, Planar numerical cosmology. II. The difference equations and numerical tests, Astrophys. J. Suppl. Ser. 54 (1984) 229–249.
[45] G. Sleijpen, H. van der Vorst, Hybrid Bi-Conjugate gradient methods for CFD problems, Comput. Fluid Dyn. Rev. 902 (4) (1995) 1–20.
[46] M. Gee, C. Siefert, J. Hu, R. Tuminaro, M. Sala, ML 5.0 Smoothed Aggregation User's Guide, Tech. Rep. SAND2006-2649, Sandia National Laboratories, 2006.
[47] M. Heroux, R. Bartlett, V. Howle, R. Hoekstra, J. Hu, T. Kolda, R. Lehoucq, K. Long, R. Pawlowski, E. Phipps, A. Salinger, H. Thornquist, R. Tuminaro, J. Willenbring, A. Williams, K. Stanley, An overview of the Trilinos project, ACM Trans. Math. Softw. 31 (3) (2005) 397–423, https://doi.org/10.1145/1089014.1089021.
[48] M.J. Abraham, T. Murtola, R. Schulz, S. Páll, J.C. Smith, B. Hess, E. Lindahl, GROMACS: high performance molecular simulations through multi-level parallelism from laptops to supercomputers, SoftwareX 1 (2015) 19–25.

[49] I.F. Sbalzarini, J.H. Walther, M. Bergdorf, S.E. Hieber, E.M. Kotsalis, P. Koumoutsakos, PPM–a highly efficient parallel particle–mesh library for the simulation of continuum systems, J. Comput. Phys. 215 (2) (2006) 566–588.
[50] B. Hess, C. Kutzner, D. Van Der Spoel, E. Lindahl, GROMACS 4: algorithms for highly efficient, load-balanced, and scalable molecular simulation, J. Chem. Theory Comput. 4 (3) (2008) 435–447, https://doi.org/10.1021/ct700301q.
[51] N.G. Deen, T. Solberg, B.H. Hjertager, Large eddy simulation of the gas liquid flow in a square cross sectioned bubble column, Chem. Eng. Sci. 56 (2001) 6341–6349.
[52] W. Harteveld, J. Julia, R. Mudde, H. van den Akker, Large scale vortical structures in bubble columns for gas fraction in the range of 5%-25%, in: Proceedings of CHISA 2004 Conference, 2004, pp. 2963–2982.
[53] M. Masterov, M.W. Baltussen, J.A.M. Kuipers, Numerical simulation of a square bubble column using Detached Eddy Simulation and Euler–Lagrange approach, Int. J. Multiph. Flow 107 (2018) 275–288.