EPJ Web of Conferences **302**, 03002 (2024) https://doi.org/10.1051/epjconf/202430203002
SNA + MC 2024

Bubble flow analysis using multi-phase field method

Kenta Sugihara[1,*], *Naoyuki* Onodera[1,**], *Yos* Sitompul[1,***], *Yasuhiro* Idomura[1,****], and *Susumu* Yamashita[2,†]

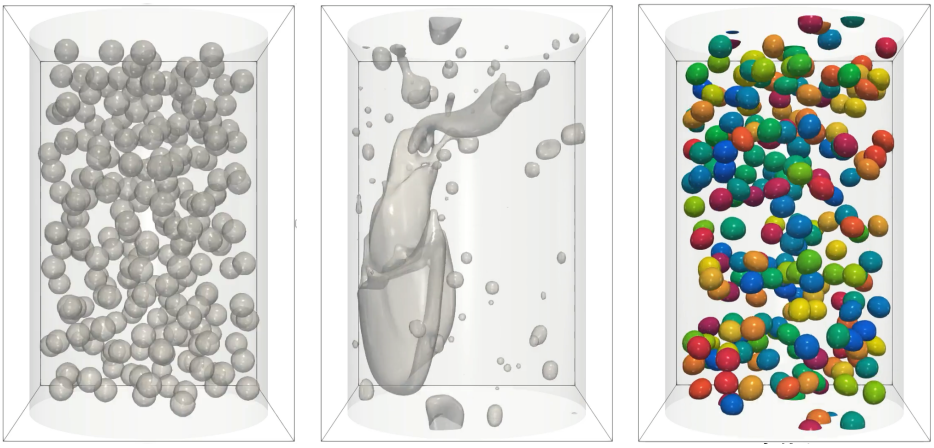
[1]Center for Computational Science & e-Systems, Japan Atomic Energy Agency
[2]Nuclear Science and Reactor Engineering Division, Nuclear Science and Engineering Center, Japan Atomic Energy Agency

Abstract. In simulations of gas-liquid two-phase flows using conventional interface capture methods, we observed that when bubbles come close to each other, they tend to merge numerically, despite experimental evidence indicating that they would repel each other. Given the significant impact of sequential numerical coalescence on flow patterns, it is necessary to regulate the merging behavior of close bubbles. To address this issue, we introduced the Multi-Phase Field (MPF) method, which mitigates bubble coalescence by applying an independent fluid fraction function to each bubble. In this study, we employed the MPF based on the N-phase model [7] to minimize numerical errors associated with surface interactions at triple junction points. Additionally, we implemented the Ordered Active Parameter Tracking (OAPT) method [9] to efficiently store several hundreds of fluid fraction functions. To validate the MPF method, we conducted analysis of turbulent bubbly pipe flows and compared the results against experimental data from Colin et al [12]. The validation results showed reasonable agreements with respect to the bubble distribution and the flow velocity profiles.

1 Introduction

In nuclear engineering, the analysis of gas-liquid two-phase flows plays an important role in core design and safety evaluation. The interface capture approach and the interface tracking approach are widely used in simulations of gas-liquid two-phase flows, and enable the analysis of complex motions of the gas-liquid interface. Unlike the interface tracking approach such as the front tracking method, the interface capture approach does not require explicit treatment of geometric changes such as coalescence and separation of interfaces, and thus, its numerical methods are simple. Two widely used numerical methods are the Volume of Fluid (VOF) method, which guarantees the volume conservation, and the Level Set (LS) method, which has better smoothness of interface shape. In addition, recently, the diffuse interface method such as the Phase Field (PF) method, which satisfies both the volume conservation and smooth interface shape, was proposed for gas-liquid two-phase flow simulations.

*e-mail: sugihara.kenta@jaea.go.jp
**e-mail: onodera.naoyuki@jaea.go.jp
***e-mail: sitompul.yos@jaea.go.jp
****e-mail: idomura.yasuhiro@jaea.go.jp
†e-mail: yamashita.susumu@jaea.go.jp



(a) Initial bubble profile (b) Single Phase Field (c) Multi-Phase Field

Figure 1: Example of bubble flow calculation for a 4.2 mm size bubble flowing in a 40 mm diameter pipe ($D_b = 10\Delta x$).

However, a critical limitation of the interface capture approach is its tendency for interfaces to numerically merge when they are within close proximity, typically within about three cells. This poses a significant challenge in accurately predicting the flow patterns of gas-liquid two-phase flows with many bubbles. According to Zhang et al.[1], the mesh resolution near the interface must be less than 1/1600 of the bubble diameter to reproduce the experimentally observed repulsion between bubbles in calculations, but it is difficult to perform such a high-resolution calculation in most practical problems. To resolve this issue, we introduce the Multi-Phase Field (MPF) method, which mitigates bubble coalescence by applying an independent fluid fraction function to each bubble.

Figure 1 shows comparisons of downward bubbly flow simulations using the conventional PF method and the new MPF method. The initial condition is given based on the void fraction, the diameter of bubbles, and the pressure gradient in the experimental condition D3 of Colin et al.[12] (see Fig.1a). The PF method shows continuous coalescence of bubbles, resulting in significantly different flow patterns (see Fig.1b). On the other hand, the MPF method does not cause coalescence of bubbles, and thus, is able to keep the initial flow pattern given by the experimental data (see Fig.1c).

In this study, we develop efficient implementation of the MPF method, and enable simulations of gas-liquid two-phase flows with many bubbles. We validate the developed simulation by computing turbulent bubbly pipe flows.

2 Multi-Phase Field method

The MPF method was originally developed for metallic crystal growth calculations, but has recently been used as one of the interface capture approach for multi-phase flow calculations. The MPF method based on the conservative Allen-Cahn (CAC)[3] equation was developed by Aihara et al. [5][6], and the N-phase model was proposed as its variant for solving the triple junction points problem [7]. In this study, we use the N-phase model.

2.1 Phase Field method

The phase field variable or the fluid fraction function ϕ in the CAC equation smoothly and steeply connects the gas phase with $\phi = 0$ to the liquid phase with $\phi = 1$ across the interface defined at $\phi = 0.5$. The governing equation is described as

$$\frac{\partial \phi}{\partial t} + \nabla \cdot (\mathbf{u}\phi) = \nabla \cdot \left(\gamma \varepsilon (\nabla \phi) - \gamma \left(\phi (1 - \phi) \frac{\nabla \phi}{|\nabla \phi|} \right) \right), \tag{1}$$

where $\mathbf{u}$ is the velocity vector describing the movement of the interface, γ is the phase field parameter describing the intensity of the phase field model, and ε is a constant related with the relative magnitudes of the diffusion and anti-diffusion terms in the phase field model.

2.2 Mirjalili's N-phase model

Mirjalili et al. developed a variant of the MPF method for an N-phase system[7]. This method can satisfy the volume conservation, the symmetry with respect to the phases, and the so-called reduction consistency, which means that in the absence of M phases, the model reduces to that for N-M phases. The N-phase model is described as

$$\frac{\partial \phi_p}{\partial t} + \nabla \cdot (\mathbf{u}\phi_p) = \nabla \cdot \left(\gamma \varepsilon (\nabla \phi_p) - \gamma \sum_{q \neq p} \phi_p \phi_q \mathbf{n}_{pq} \right) \quad \text{for} \quad 1 \leq p \leq N, \tag{2}$$

where p and q are the phase indices, N is the number of phases, and $\mathbf{n}_{pq}$ is the pairwise normal vector of the interface defined as

$$\mathbf{n}_{pq} = \frac{\nabla \phi_{pq}}{|\nabla \phi_{pq}|} \quad \text{for} \quad p \neq q, \tag{3}$$

calculated from the pairwise volume fraction

$$\phi_{pq} = \frac{\phi_p}{\phi_p + \phi_q} \quad \text{for} \quad p \neq q. \tag{4}$$

This method conserves not only N individual phases but also the sum of all phases ($\sum \phi_p = 1$) for each computational cell. Since $\mathbf{n}_{pq} = -\mathbf{n}_{qp}$, adding all N phases in Eq.(2) cancels the anti-diffusion term and yields

$$\frac{\partial}{\partial t} \sum_p \phi_p + \sum_p \nabla \cdot (\mathbf{u}\phi_p) = \varepsilon \sum_p \nabla \cdot (\gamma \nabla \phi_p). \tag{5}$$

Thus, if $\sum_p \phi_p = 1$ is satisfied in the initial condition, then $\sum_p \phi_p = 1$ is always satisfied in the time integration as long as linear discretization is applied to the advection and diffusion terms. Numerical schemes such as WENO and discretization using slope limiter do not satisfy the above condition. In such cases, normalization is necessary to satisfy $\sum_p \phi_p = 1$ in all computation cells. In this study, we apply the 3rd-order TVD Runge-Kutta method to the time integration, the 3rd-order MUSCLE method to the advection term, and the 2nd-order FVM method to the others. This approach satisfies the above condition and $\sum_p \phi_p = 1$ is guaranteed.

2.3 Ordered Active Parameter Tracking (OAPT) method

In a naive implementation of the MPF method, it is necessary to store an array of N phase field variables in memory and compute them for all computation cells. However, the computational cost becomes very high when calculating a large number of phases (more than several hundreds). To save the memory and the computational cost, the Active Parameter Tracking (APT) method was proposed[8]. In addition, the Ordered APT (OAPT) was developed by improving APT considering continuous memory access which is important for optimizing GPU computation[9].

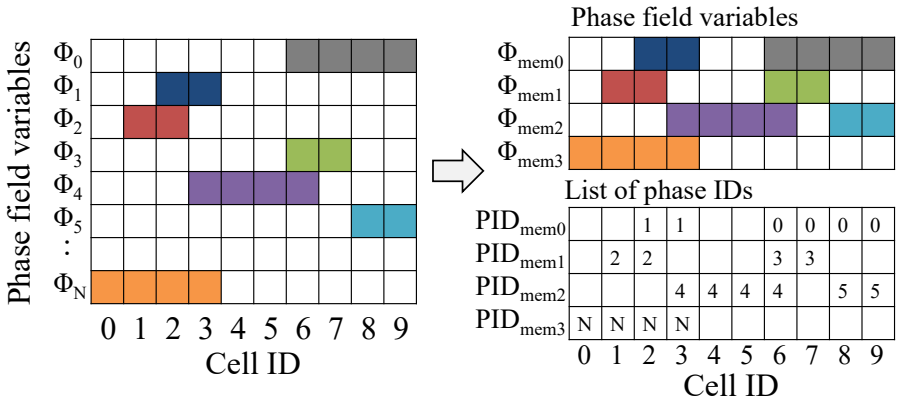


Figure 2: Ordered Active Parameter Tracking method[9].

Figure 2 (left) is an example of storing N phases in a naive implementation, where each phase is stored in each memory layer. Active cells with non-zero phase field variables are colored, and white cells are not used in the calculation, leading to waste of the memory and the computational cost. Figure 2 (right) is an example of the OAPT method, in which the array of N phase field variables is compressed by storing multiple phase field variables in the same memory layer. The phase filed variables in the same memory layer are labelled by an array for the phase index (phase ID). Re-ordering of the phase field variable is performed when active cells become close to each other in the same memory layer. This process is required about once every 500 steps in the bubbly flow simulation in Section 4.

In this study, the following processes are performed at every 10 times steps.

1. **Calculate signed distance functions**
   Create Level Set functions[2] for each memory layer up to a distance of about three cells from the interface to determine the active cells.

2. **Cutoff for phase field variables**
   Set the phase field value to 0 outside the active cells. Update active cells.

3. **Reordering**
   If the distance between active cells with different phase IDs are less than two cells, the active cells for one of these phase IDs are copied to the different memory layer so that they do not overlap each other.

4. **Normalize**
   The phase field variables are normalized so that the sum of the phases is 1 on each computational cell.

5. **Mass correction**
   The change of mass in Steps 2 and 4 is corrected in the same way as the mass redistribution method by Chiu et al.[3]. Here, reference mass is calculated prior to Step 2.

Since the equilibrium solution of the PF equation is an infinite continuous function, it is characterized by a small phase filed value that spreads as the interface moves. Therefore, if the phase field variable spreads outside the active cells, they are eliminated from the computation, so that the size of active cells is almost constant, and the OATP method works with a finite number of memory layers. In this study, the range of active cells is limited to $3\Delta x$ from the interface using a signed distance function (Step 2). This method is compatible with the OAPT method, where all cells with the same phase ID is always stored in the same memory layer, and thus, LS functions are easily computed.

3 Two-phase flow solver

The following Navier-Stokes equations are adopted as the basic equations in this study,

$$\nabla \cdot \mathbf{u} = 0, \tag{6}$$

$$\frac{\partial \mathbf{u}}{\partial t} + (\mathbf{u} \cdot \nabla)\mathbf{u} = \frac{1}{\rho}\nabla \cdot (\mu(\nabla \mathbf{u} + (\nabla \mathbf{u})^T)) - \frac{1}{\rho}\nabla p + \frac{1}{\rho}\mathbf{F}_s + \frac{1}{\rho}\mathbf{F}_b, \tag{7}$$

where $\mathbf{u}$ is the velocity, ρ is the density, μ is the viscosity, p is the pressure, $\mathbf{F}_s$ is the surface tension force, $\mathbf{F}_b$ is the body force including the gravitational force and the pressure gradient of pipe flows.

The phase field variable, which is computed using the PF or MPF method, is used not only for identifying the interface but also for defining the physical properties of gas-liquid two-phase flows. In this study, the fluid fraction variable ϕ is computed in the same way as Balcazar et al.[10],

$$\phi(\mathbf{x}, t) = 1 - max(\phi_1(\mathbf{x}, t), \ldots, \phi_{N-1}(\mathbf{x}, t), \phi_N(\mathbf{x}, t)), \tag{8}$$

where $\phi_1, \ldots, \phi_N$ are the phase field variables of N bubbles, respectively. This method can define a thin fluid fraction between bubbles when they are in contact with each other. This enables accurate computation of the pressure and velocity fields at the interface, and mitigates spurious bubble coalescence.

The density and the viscosity are defined as

$$\rho = \rho_l \phi + \rho_g (1 - \phi), \tag{9}$$

$$\mu = \mu_l \phi + \mu_g (1 - \phi), \tag{10}$$

where ρ_l, ρ_g are the density of liquid and gas phases, and μ_l, μ_g are the viscosity of liquid and gas phases, respectively.

The continuum surface force (CSF) model is applied to the surface tension force $\mathbf{F}_s$ as

$$\mathbf{F}_s = \sigma \sum_i \kappa_i \nabla \phi_i, \tag{11}$$

where σ is the surface tension coefficient, and κ_i and ϕ_i are the curvature and the phase field variable of the i-th bubble, respectively.

For the calculation of incompressible fluids, the SMAC method is applied to correct the pressure and velocity fields. The pressure Poisson equation is the most costly part of incompressible fluid calculations, especially for gas-liquid two-phase flow simulations with large density contrasts, which lead to ill-conditioned matrices. A cache-reuse multi-grid preconditioned conjugate gradient (CRMG-CG) method [11] is applied to solve the pressure Poisson equation. The third-order TVD Runge-Kutta method was applied to the time integration of the explicit parts of the Navier-Stokes equation (7) and the MPF equation (2).

4 Turbulent bubbly downward flows

The two-phase flow solver with the MPF method is validated in simulations of turbulent bubbly pipe flows at high density ratios and high Reynolds numbers.

4.1 Problem settings

Turbulent bubbly flows were simulated in a pipe with diameter of $D = 40$ mm. Following the experimental condition D3[12], air and water were set for the gas-liquid phases, bubbles with an average void fraction of 7.5% and bubble diameter of 4.2 mm were randomly placed in the initial condition (see Fig.1a), and the downward flow driven by the pressure gradient of $\frac{dP}{dz} = 8662$ Pa/m was analyzed in the computational domain of $L_x \times L_y \times L_z = D \times D \times \pi D$, where (x, y, z) gives Cartesian coordinates, the grid size is $dx = D/192$, and the periodic boundary condition is applied in the vertical z direction. The number of bubbles is $N_b = 153$.

The external force in the Navier-Stokes equation (7) is defined as

$$\mathbf{F}_b = \beta\, \hat{\mathbf{z}} + (\rho - \bar{\rho})\, \mathbf{g}, \tag{12}$$

where $\hat{\mathbf{z}}$ is the unit normal vector in the z direction, and $\bar{\rho}$ is the average mixture density in the pipe, and $\mathbf{g}$ is the gravitational acceleration vector, and β is defined as

$$\beta = \frac{dP}{dz} + \bar{\rho} g, \tag{13}$$

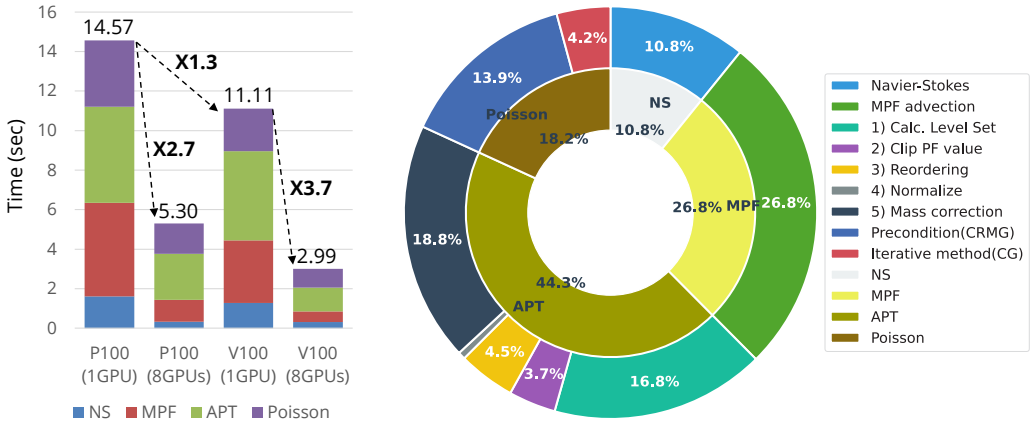
where $g = -9.8 m/s^2$ is the gravitational acceleration.

4.2 Computational environment and cost

The calculations were performed using eight NVIDIA V100 GPUs on NVIDIA DGX-2[14] and NVIDIA P100 GPUs on TSUBAME3.0[13].

Figure 3b shows the breakdown of computation time excluding data output in the initial time step, which shows that APT accounts for 44.3% of the total time. APT is performed once every 10 steps, and thus, its average cost becomes 1/10. APT is dominated by the Level Set calculation and the mass correction. The Level Set calculation is costly because it requires iterative processes for all memory layers. The mass correction occupies a high cost for executing many reductions by the number of phases ($N_b = 153$).

Figure 3a shows a comparison of computational performance using NVIDIA P100 GPUs on TSUBAME3.0 and NVIDIA V100 GPUs on DGX-2. For single GPU calculations, calculations using V100 GPU are $\times 1.3$ faster than those using P100 GPU. Parallel computation using eight V100 GPUs further improves the performance by $\times 3.7$.



(a) Strong scaling test.

(b) Breakdown of computation time(V100, 1GPU).

Figure 3: Measurement of computation time for calculations with $192 \times 192 \times 288$ cells.

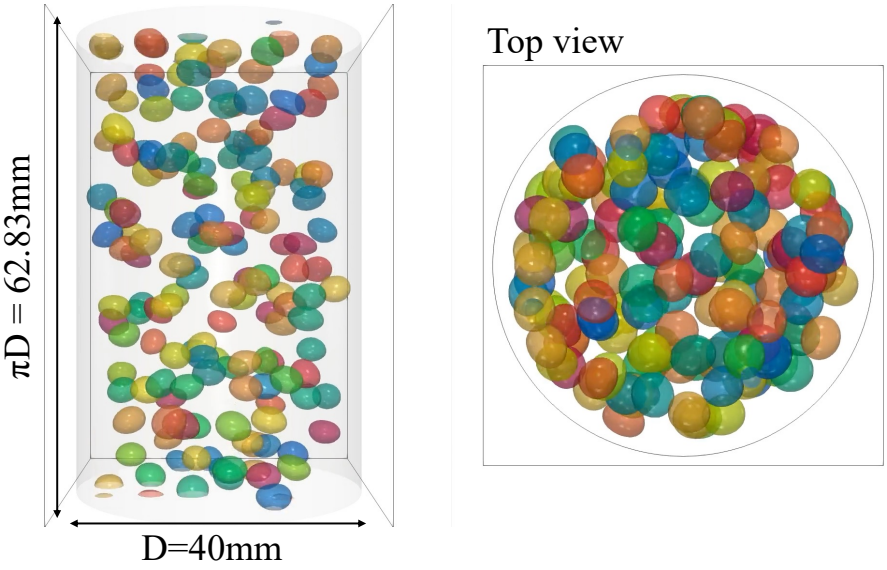
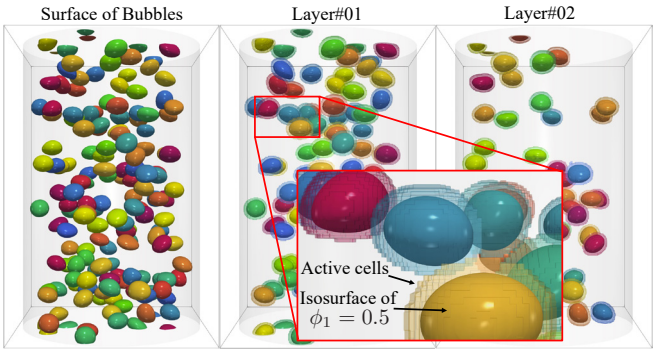


Figure 4: Bubble surfaces colored by the phase IDs.

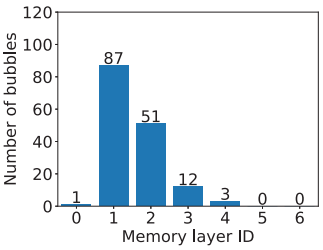
4.3 Simulation results

Figure 4 shows a snapshot of a bubble interface colored by the phase IDs. The MPF method makes it possible to calculate a large number of bubbles, which are densely packed together. This kind of simulation was difficult with the conventional interface capture methods. Visualization of the gas-liquid interface using the ray tracing technique is also shown in Fig.7.

Figure 5 shows the storage of each phase data using the OAPT method. In this calculation, the zeroth memory layer was used for the liquid phase, and the remaining six memory

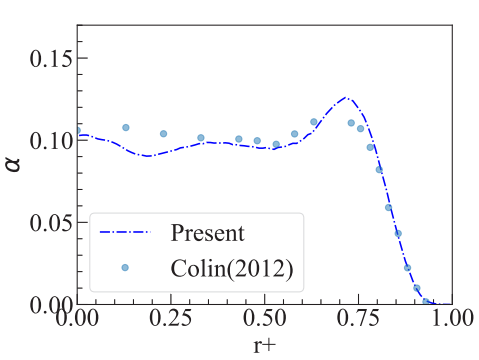


(a) Storage of phase field variables at active cells.

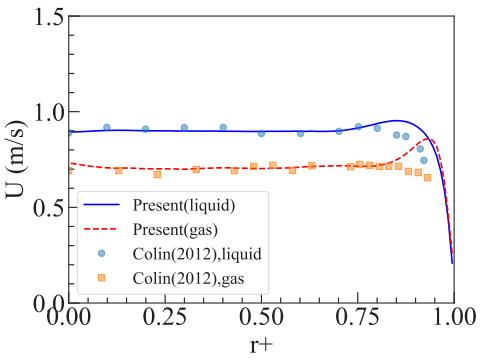


(b) Number of bubbles stored in each memory layer.

Figure 5: View of 153 bubbles data storage.



(a) Void fraction.



(b) Flow velocity.

Figure 6: Radial distributions of the void fraction and the flow velocity.

layers were prepared for the gas phase to store 153 bubbles, while only four memory layers contained the gas phase data. This means that in the current simulation, the maximum number of bubbles in contact with one bubble is three. This results shows that the OATP method reduces the memory size to 7/153 compared to a naive implementation. Since the OAPT method is used, stencil calculations in each memory layer can be performed as in interface advection calculations in the conventional PF method. As can be seen in the enlarged image, active cells are defined up to a computational cell $3\Delta x$ away from the surface, and each active cell is controlled so that they do not touch each other.

Figure 6 shows the radial distributions of the void fraction α and the flow velocity U, where $r+$ is the normalized radius and the distribution is averaged over z and an annulus at $r+$. It is known that bubbles tend to be gathered in the center of the pipe in downward flows due to the balance of wall shear stress, pressure gradient, and gravity. The present calculation reproduces this tendency, and the bubble distribution shows quantitative agreement with the experiment (see Fig.6a). The flow velocity profiles also show reasonable agreements, reproducing the flow velocities of gas and liquid phases due to buoyancy (see Fig.6b).

Figure 7: Visualization of bubble flow with ray tracing.

5 Summary

We developed a new two-phase flow solver by implementing the MPF method based on Mirjalili's N-phase model using the OAPT method on GPU supercomputers. The developed solver was applied to turbulent bubbly pipe flows, and was validated against the experiment by Colin et al. The results showed reasonable agreements with respect to the bubble distribution and the flow velocity profiles of gas and liquid phases. By applying the OAPT method, 153 bubbles or phases data were stored only in six memory layers, and the memory size and the computational cost were reduced by an order of magnitude compared to a naive implementation.

Acknowledgments

This research was partly supported by JSPS KAKENHI, Grant-in-Aid for Scientific Research (C) JP24K14973, and "Joint Usage/Research Center for Interdisciplinary Large-scale Information Infrastructures (JHPCN)" in Japan (Project ID: jh240071). Numerical computation in this work was partially carried out on the supercomputer HPE SGI8600 at the Japan Atomic Energy Agency, and TSUBAME3.0 at the Tokyo Institute of Technology.

References

[1] Jie Zhang et al., Physical Review Fluids **4**, 043604 (2019).
[2] Mark Sussman et al., J. Comput. Phys. **114**, 146-159 (1994).
[3] Pao-Hsiung Chiu and Yan-Ting Lin, J. Comp. Phys. **230**, 185-204 (2011).
[4] Kenta Sugihara et al., JAEA-Research-2023-006, (2023).
[5] Shintaro Aihara et al., Computers and Fluids **178**, 141-151 (2019).
[6] Shintaro Aihara et al., Theor. Comput. Fluid Dyn. **37**, 639-659 (2023).
[7] Shahab Mirjalili and Ali Mani, Journal of Computational Physics **498**, 112657 (2024).
[8] Akinori Yamanaka et al., J. Comput. Sci. Technol. **6** (3) 182–197 (2012).
[9] Yos Sitompul et al., Journal of Computational Science **64**, 101832 (2022).
[10] Nestor Balcazar et al., Int. J. Multiphase Flow **74**, 125-142 (2015).
[11] Naoyuki Onodera et al., HPC Asia 2021, 120-128 (2021).
[12] Colin et al., Journal of Fluid Mechanics **711**, 461-515 (2012).
[13] TSUBAME3.0, https://www.gsic.titech.ac.jp/sites/default/files/spec30e_0.pdf, (2017)
[14] NVIDIA DGX-2, https://www.nvidia.com/content/dam/en-zz/Solutions/Data-Center/dgx-1/dgx-2-datasheet-us-nvidia-955420-r2-web-new.pdf, (2019)