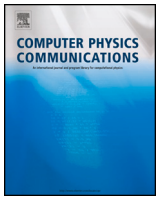
Computer Physics Communications 295 (2024) 109012

Contents lists available at ScienceDirect

Computer Physics Communications

journal homepage: www.elsevier.com/locate/cpc



Computational Physics

A single-phase GPU-accelerated surface tension model using SPH ☆

Chunze Cen *, Georgios Fourtakas, Steven Lind, Benedict D. Rogers

School of Engineering, Faculty of Science and Engineering, University of Manchester, Manchester M13 9PL, UK

ARTICLE INFO

Keywords:
Smoothed particle hydrodynamics
SPH
Surface tension model
GPU
Droplets
DualSPHysics

ABSTRACT

This paper presents an accelerated single-phase surface tension smoothed particle hydrodynamics (SPH) solver developed to run entirely on a graphics processing unit (GPU) capable of simulating millions of particles in three dimensions on a single GPU. The single-phase surface tension model is augmented with a contact line force to improve the prediction of the physics at the liquid-solid contact point. The surface tension model uses the modified dynamic boundary condition (mDBC) to impose no-slip conditions at the wall boundary. To enable simulations with millions of particles, the single-phase surface tension model has been implemented in the open-source SPH code DualSPHysics to exploit the GPU acceleration, paying special attention to the size of the kernel support and integration of the neighbour lists. The new scheme is validated using 2-D and 3-D test cases including drop deformation, drop oscillation, Rayleigh-Plateau instability and surface contact angles. The results show a good agreement with the analytical solutions with a standard spatial convergence behaviour. Profiling the new surface tension solver shows an additional computational complexity. The performance analysis shows that the new code has a speed up of up two orders of magnitude (x70-80) compared to the CPU-only code. Profiling the new CUDA kernels shows they have the near identical performance metrics with the main CUDA kernels in the original DualSPHysics solver.

1. Introduction

Surface tension is commonly found in a wide range of natural phenomena and industrial applications. The unbalanced forces of the molecules at the interface results in the surface tension force. The surface tension force is one of the most important factors determining the behaviour of drops and bubbles. It can be neglected in many large length-scale problems but it is of great importance to the applications with sufficiently small characteristic length-scales such as microfluidic devices [3], inkjet printing [28], cavitation [6], drop dynamics [20,40,52] and spray behaviours in the engine [9].

Computational fluid dynamics (CFD) has proven its ability to use multi-phase surface tension models [38] couple with the interface tracking techniques such as volume-of-fluid (VOF) [17] and level-set (LS) [43]. On the other hand, in many applications modellers are often interested in the behaviour of the flow of a single phase where the surface tension plays an important role and is essential for the flow behaviour. In addition, the single-phase model saves a large amount of computational costs compared to the multi-phase model as calculations among other phases are neglected.

An alternative to mesh-based methods is to use a meshless method such as smoothed particle hydrodynamics (SPH). SPH is a Lagrangian mesh-free method first proposed to solve non-axisymmetric astrophysical applications [14,25]. In the past two decades there has been significant progress developing SPH for engineering applications. It is now approaching a mature status as a CFD method with decades of development due to its advantages for free-surface flows over the traditional mesh-based CFD methods [26,46,49]. In comparison to the aforementioned mesh-based schemes, SPH offers unique advantages as the interface between different phases or components can be captured precisely through the use of the particles with different identities.

There have been multiple formulations proposed to include surface tension on SPH. Tartakovsky and Meakin [44] proposed the pairwise force (PF) model where the surface tension is regarded as an imbalance of molecular forces at the interface. This model is intuitive and easy to implement but it requires calibration on a case-by-case basis. Various researchers proposed different formulations for the inter-particle force which is attractive over a long distance and repulsive over a short distance to obtain more accurate results [2,22,53]. Subsequently, Tartakovsky and Panchenko [45] linked the macroscopic parameters such

☆ The review of this paper was arranged by Prof. Peter Vincent.

* Corresponding author.
E-mail address: chunze.cen@postgrad.manchester.ac.uk (C. Cen).

https://doi.org/10.1016/j.cpc.2023.109012
Received 8 May 2023; Received in revised form 6 October 2023; Accepted 8 November 2023
Available online 14 November 2023

as the surface tension coefficient and the contact angle to the PF model to avoid the need for calibration but the model still resulted in non-physical pressure being generated on the interface. From the macroscopic view of surface tension, the continuum surface force (CSF) model [4], where the surface tension is rewritten into a volume force, was first tested by Morris [31] for multi-phase SPH. However, the curvature has proven difficult to predict due to the divergence of the normal direction on interface [18]. Hu and Adami [18] modified the formulations using the colour function to represent different phases and their gradients to compute normals and the interface curvature. Adami et al. [1] extended the model to be more accurate by using a weight colour function and a reproducing divergence approximation for the curvature. Following Adami et al. [1], Zöller et al. [54] proposed the density partitioned continuous surface-stress which could accurately predict the surface tension force with complex physics and is stable at a large time step for high density and viscosity ratio. The above-mentioned improvements of the CSF model are all related to a multi-phase SPH. This is computationally expensive since a large number of SPH particles are required to represent multiple phases. There have been some single-phase surface tension models developed using SPH [11,37,42]. Recently, Vergnaud et al. [47] optimized the CSF model to be applied to the single-phase SPH and various modifications have been made to improve its accuracy with robust and accurate results. This avoids the computation expense of the larger number of particles required for the multi-phase approach. They also proposed a normal vector correction approach by enforcing the fluid to the equilibrium contact angle. In addition to the enforced normal vector approach, Huber et al. [19] derived a contact line force model based on the difference between dynamic contact angle and equilibrium contact angle, providing a more natural method of capturing the dynamic wetting behaviour, which is of great importance in the dynamic behaviour of the droplet surface interaction applicable to various research areas such as drop manipulation.

DualSPHysics is an accurate and robust open-source solver based on SPH method to study free-surface flows [8,10]. The multi-phase model of DualSPHysics has implemented a surface tension model proposed by Hu and Adami [18], but the single-phase model still lacks an implementation. There are many single-phase applications that would benefit from the inclusion of a surface tension model including bubble coalescence [7], flow in the micro-channels [16] and heat transfer [50]. Given that the single-phase model requires less computation resource, it is necessary to select and implement a single-phase surface tension model in a GPU-accelerated code.

This paper aims to develop an enhanced GPU-accelerated single-phase surface tension model as initially proposed by Vergnaud et al. [47] in the absence of the stabilising Riemann solver formulation which is not conservative, by combining the contact line force model from Huber et al. [19] in conjunction with the modified dynamic boundary condition (mDBC) no-slip wall boundary condition of English et al. [12]. The surface tension model is implemented in the DualSPHysics solver following rigorous validation of the model for 2-D and 3-D test cases. There are multiple challenges in developing the single-phase surface tension model for GPU implementation. Firstly, the new algorithms will have an additional computational overhead when integrated into a pre-existing GPU-accelerated code, requiring special attention when using the neighbour lists efficiently while maximising GPU occupancy. Furthermore, there is the additional complication of how to use mixed precision variables on GPUs within the new model and understanding how this affects accuracy and how to achieve the maximum speed up given the extra computations required. This is particularly relevant to methods which invert matrices local to each particle as required for the eigenvalues in the surface tension model of Vergnaud et al. [47] Further and as discussed above, the new model must ensure that the single-phase surface tension model captures all the physical processes of the applications.

This paper is organized as follows: In Section 2, the governing equations and the formulations for surface tension are introduced. In Section 3, the modifications made to the DualSPHysics code are described. This is followed by a number of test cases to validate the surface tension model including drop deformation, drop oscillation, Rayleigh-Plateau instability, and contact angle in Section 4. Finally, the new CUDA kernels that have been implemented in the code are profiled and the results are compared to the original version of DualSPHysics using Nsight Compute, a CUDA kernel profiler.

2. Numerical methods

2.1. SPH fundamentals

In SPH, the value of a field function $f(\mathbf{x})$ at point $\mathbf{x}$ is approximated by a local convolution of the function and a weighting kernel over the supporting domain expressed as,

$$< f(\mathbf{x}) > = \int_{\Omega} f(\mathbf{x}')W(\mathbf{x}-\mathbf{x}',h)d\mathbf{x}' \tag{1}$$

where Ω is the supporting domain, W denotes the smoothing kernel, h is a characteristic length called the smoothing length and $< ... >$ denotes the continuous SPH interpolation.

The discretised form of Eq. (1) is written as,

$$f(\mathbf{x}_i) = \sum_j \frac{m_j}{\rho_j} f(\mathbf{x}_j)W(\mathbf{x}_i-\mathbf{x}_j,h), \tag{2}$$

where the subscripts i and j refer to the ith and jth particle, m is the mass, ρ the density and the $< ... >$ has been dropped for simplicity.

To simplify the notation, Eq. (2) can be rewritten as,

$$f(\mathbf{x}_i) = \sum_j f_j W_{ij} V_j, \tag{3}$$

where $f_j = f(\mathbf{x}_j)$ is the value of function f on the jth particle, $V_j = \frac{m_j}{\rho_j}$ is the volume of the jth particle and $W_{ij} = W(\mathbf{x}_i-\mathbf{x}_j,h)$ is the kernel evaluated between particles i and j.

Similarly, in continuous form, the gradient operator in SPH is written as,

$$< \nabla f(\mathbf{x}) > = \int_{\Omega} f(\mathbf{x}')\nabla W(\mathbf{x}-\mathbf{x}',h)d\mathbf{x}'. \tag{4}$$

This can be discretized as

$$\nabla f(\mathbf{x}_i) = \sum_j f_j \nabla_i W_{ij} V_j. \tag{5}$$

Equation (5) is the most basic form of SPH gradient. A combination of other expressions is used in practice to ensure mass, momentum and energy conservation [29,30,48].

The quintic Wendland kernel [51] is used in this study due to its good robustness for the tensile instability [27]. The kernel is given by,

$$W_{ij} = \alpha_D(1-\frac{r_{ij}}{2h})^4(1+\frac{2r_{ij}}{h}) \quad \text{for} \quad 0 \le r_{ij} \le 2h \tag{6}$$

where $r_{ij} = |\mathbf{r}_i - \mathbf{r}_j|$ denotes the distance between particle i and j and α_D is the normalization term for different dimensions where $\alpha_D = \frac{7}{4\pi h^2}$ in 2-D and $\alpha_D = \frac{21}{16\pi h^3}$ in 3-D.

2.2. Governing equations

The Lagrangian form of conservation of mass and momentum is written as,

$$\begin{aligned} \frac{D\rho}{Dt} &= -\rho\nabla\cdot\mathbf{u}, \\ \frac{D\mathbf{u}}{Dt} &= -\frac{1}{\rho}\nabla p + \nu\nabla^2\mathbf{u} + \mathbf{f}_{body}, \end{aligned} \tag{7}$$

where $\mathbf{u}$ is the velocity, p denotes the pressure, ν represents the kinematic viscosity and $\mathbf{f}_{body}$ is the body force including surface tension force, contact line force and gravity.

2.3. SPH discretisation

Based on DualSPHysics formulation [10] and by replacing the viscous term from Morris et al. [32], the corresponding SPH discrete form of Equation (8) is given by

$$\begin{aligned}\frac{D\rho}{Dt} &= \rho_i \sum_j \frac{m_j}{\rho_j}\mathbf{u}_{ij}\cdot\nabla_i W_{ij},\\ \frac{D\mathbf{u}_i}{Dt} &= -\sum_j m_j\left(\frac{P_i+P_j}{\rho_i\rho_j}\right)\nabla_i W_{ij}\\ &\quad+\sum_j \frac{m_j(\nu_i+\nu_j)\mathbf{r}_{ij}\cdot\nabla_i W_{ij}}{\rho_j(|r_{ij}|+0.001h^2)}+\mathbf{g}+\mathbf{f}_s+\mathbf{f}_{clf},\end{aligned} \tag{8}$$

where $\mathbf{g}$ is the gravitational force, $\mathbf{f}_s$ denotes the surface tension force and $\mathbf{f}_{clf}$ represents the contact line force. The term $0.001h^2$ aims to prevent the singularity when r_{ij} tends to zero.

The Tait equation of state is used to couple the density and pressure as follows,

$$p = \frac{\rho_0 c_0^2}{\gamma}\left[\left(\frac{\rho_i}{\rho_0}\right)^{\gamma}-1\right] \tag{9}$$

where ρ_0 is the reference density, c_0 the speed of sound and γ the polytropic index which is taken as 7 for water.

2.4. Surface tension model

The continuum surface force (CSF) model is a widely used surface tension model in numerical simulation of fluid dynamics [4]. In the CSF model, the surface tension force per unit volume $\mathbf{F}_s$ is transferred from a surface force to a volume force [4],

$$\mathbf{F}_s = \mathbf{f}_s\delta_s, \tag{10}$$

where δ_s is a normalized function and $\mathbf{f}_s$ is the force per unit area given as,

$$\mathbf{f}_s = -\sigma\kappa\hat{\mathbf{n}}, \tag{11}$$

where σ is the surface tension coefficient, κ denotes the curvature of the interface and $\hat{\mathbf{n}}$ is the unit normal vector to the interface, where the hat symbol denotes the normalised vector.

In this work, a recently proposed single phase corrected-CSF (C-CSF) model [47] has been implemented in DualSPHysics and extended with a contact line model of Huber et al. [19] to capture the physics of liquid-solid contact line more accurately.

2.5. Single-phase surface tension SPH formulation

According to the CSF model [4], the surface tension force $\mathbf{F}_{s,i}$ for each particle i could be written as,

$$\mathbf{F}_{s,i} = -\sigma\kappa_i\hat{\mathbf{n}}_i\delta_{s,i}, \tag{12}$$

where the σ is the surface tension coefficient, κ_i is the curvature of the interface, $\mathbf{n}_i$ denotes the normal to the interface and $\delta_{s,i}$ presents the interfacial function. The unit vector normal to the interface from Vergnaud et al. [47] is expressed as,

$$\hat{\mathbf{n}}_i = -\frac{\nabla\lambda_i}{\|\nabla\lambda_i\|}, \tag{13}$$

where

$$\nabla\lambda_i = \sum_j(\lambda_j-\lambda_i)(\mathbb{L}_i\nabla W_{ij})V_j, \tag{14}$$

and λ_i is the minimum eigenvalue of the matrix $\mathbb{L}_i^{-1}$. The $\mathbb{L}_i^{-1}$ matrix is the inverse of correction matrix $\mathbb{L}_i$ [47] defined as,

$$\mathbb{L}_i^{-1} = \sum_j(\mathbf{r}_j-\mathbf{r}_i)\otimes\nabla_i W_{ij}V_j. \tag{15}$$

The surface curvature of particle i can be evaluated as,

$$\kappa_i = \sum_j \mathbb{L}_i(\hat{\mathbf{n}}_j-\hat{\mathbf{n}}_i)\cdot\nabla W_{ij}V_j. \tag{16}$$

Generally, in the multi-phase CSF models [4], the interfacial function $\delta_{s,i}$ is usually determined by the gradient of colour function ∇c_i defined as,

$$\nabla c_i = \sum_j(c_i+c_j)\nabla W_{ij}V_j, \tag{17}$$

where c_i and c_j are values of the colour function on particle i and j. In the single-phase model, the interfacial function is expressed as Equation (18) due to $c_i = c_j = 1$.

$$\delta_{s,i} = \|\nabla c_i\| = 2\|\nabla W_{ij}V_j\|. \tag{18}$$

One challenge for a single-phase model is that the surface particle has a truncated kernel support with less neighbouring particles which results in increased error. Several corrections have been proposed to improve the accuracy in [47]. A coefficient $\lambda_{threshold}$ is used to test if the particle i has sufficient neighbouring particles, where the gradient of the normal, $\nabla\lambda_i$, is modified as,

$$\nabla\lambda_i = \begin{cases}\sum_j(\lambda_j-\lambda_i)(\mathbb{L}_i\nabla W_{ij})V_j & \text{if } \lambda_i \geq \lambda_{threshold}\\ \sum_j\lambda_j(\mathbb{L}_i\nabla W_{ij})V_j & \text{otherwise,}\end{cases} \tag{19}$$

where $\lambda_{threshold}$ is 0.7 in all cases.

For the particles located far from the surface, the value of $\nabla\lambda_i$ tends to zero. Therefore, to ensure that $\nabla\lambda_i$ on these particles is exactly zero and to reduce the computational cost, the normal is modified as,

$$\hat{\mathbf{n}}_i = \begin{cases}-\frac{\nabla\lambda_i}{||\nabla\lambda_i||} & \text{if } ||\nabla\lambda_i|| > \epsilon\frac{\lambda_i}{h}\\ 0 & \text{otherwise,}\end{cases} \tag{20}$$

where h is the constant smoothing length and ϵ is a constant and is taken as 0.1 herein following Vergnaud et al. [47]. Similarly, a correction for the curvature is applied as,

$$\kappa_i = \sum_j s_{ij}(\hat{\mathbf{n}}_j-\hat{\mathbf{n}}_i)\cdot(\mathbb{L}_i\nabla W_{ij})V_j, \tag{21}$$

where

$$s_{ij} = \begin{cases}1 & \text{if } \hat{\mathbf{n}}_i\cdot\hat{\mathbf{n}}_j \geq \cos\alpha_t \text{ and } ||\hat{\mathbf{n}}_i|| > 0 \text{ and } ||\hat{\mathbf{n}}_j|| > 0\\ 0 & \text{otherwise,}\end{cases} \tag{22}$$

where α_t is the threshold defined as $\cos\alpha_t = -\frac{1}{2}$ and $\cos\alpha_t = -\frac{1}{3}$ in 2D and 3D, respectively. When considering the interfacial function, some particles located at the vertex of the geometry moving away from the fluid domain due to the singularity are observed in some simulations. To solve this problem, the following modification [47] was proposed:

$$\delta_{s,i} = 2\max(1,\phi_i)||\sum_j\nabla W_{ij}V_j||, \tag{23}$$

where

$$\phi_i = \frac{1}{2\sum_j W_{ij}V_j}. \tag{24}$$

This correction increases slightly the surface tension force on the aforementioned detached particles thereby preventing them from escaping in an unphysical manner.

For the cases including solid boundaries, the evaluation of the normal vector near the solid boundary should be corrected for better accu-

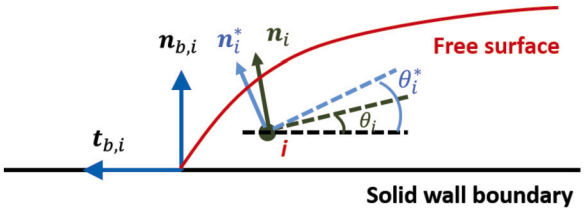


Fig. 1. Definition sketch of the normal correction procedure near the contact line, adapted from [47].

racy. The definition sketch of the normal correction procedure near the contact line, which is adapted from [47], is shown in Fig. 1. Based on [47], the tangent vector $\mathbf{t}_{b,i}$ to the boundary which is also normal to the contact line is expressed as,

$$\mathbf{t}_{b,i} = \frac{\hat{\mathbf{n}}_i - (\hat{\mathbf{n}}_i \cdot \hat{\mathbf{n}}_{b,i})\hat{\mathbf{n}}_{b,i}}{||\hat{\mathbf{n}}_i - (\hat{\mathbf{n}}_i \cdot \hat{\mathbf{n}}_{b,i})\hat{\mathbf{n}}_{b,i}||}, \tag{25}$$

where $\mathbf{n}_{b,i}$ is the approximate unit wall normal pointing away from the wall towards the fluid and computed for near-wall fluid particle i via summation over the kernel gradient of nearby wall particles j [5],

$$\hat{\mathbf{n}}_{b,i} = \frac{\sum_j \nabla W_{ij}}{|\sum_j \nabla W_{ij}|}. \tag{26}$$

Then, the corrected normal $\hat{\mathbf{n}}^*$ for particle i is defined as,

$$\hat{\mathbf{n}}_i^* = \cos\theta_i^* \hat{\mathbf{n}}_{b,i} + \sin\theta_i^* \mathbf{t}_{b,i}, \tag{27}$$

where θ_i^* is the corrected contact angle and expressed as,

$$\theta_i^* = \begin{cases} \theta_e + (\theta_i - \theta_e)(d_i/R)^2 & \text{if } d_i < R \\ \theta_i \quad \text{otherwise,} \end{cases} \tag{28}$$

where $R = 2h$ is the kernel radius, d_i is the distance from fluid particle i to the nearest boundary, θ_e denotes the equilibrium contact angle and θ_i is the contact angle before correction defined as,

$$\theta_i = \arccos(\hat{\mathbf{n}}_i \cdot \hat{\mathbf{n}}_{b,i}). \tag{29}$$

During the simulation, the computed normal $\hat{\mathbf{n}}_i$ is replaced by the corrected $\hat{\mathbf{n}}_i^*$ before the computation of curvature.

Note that, in Vergnaud et al. [47], a Riemann solver formulation was necessary to maintain numerical stability in the droplet surface interaction. Herein, we forego a Riemann solver by using a contact line force model, the wall boundary conditions of English et al. [12] and a particle shifting technique to maintain a uniform particle distribution as detailed in the following sections.

2.6. Contact line force (CLF) model

Similar to the surface tension force, the volume formation of the contact line force $\mathbf{F}_{clf}$ is given as [19]:

$$\mathbf{F}_{clf} = \mathbf{f}_{clf}\delta_{clf}, \tag{30}$$

with

$$\mathbf{f}_{clf} = \sigma\left[cos(\theta_{eq}) + \hat{\mathbf{d}}_i \cdot \hat{\mathbf{n}}_i\right]\hat{v}_i, \tag{31}$$

where the distance vector $\mathbf{d}_i$ is written as:

$$\mathbf{d}_i = \sum_{j_b} V_{j_b}\mathbf{r}_{j_b i} W_{i j_b}, \tag{32}$$

where the subscript j_b implies the summation is only applied over the boundary particles. The term v_i is computed as:

$$v_i = |\mathbf{d}_i|^2\mathbf{n}_i - (\mathbf{d}_i \cdot \mathbf{n}_i)\mathbf{d}_i. \tag{33}$$

The interfacial function δ_{clf} in (30) is given as:

$$\delta_{clf_i} = -2\hat{\mathbf{d}}_i \cdot \sum_j V_j(\delta'_{clf_j} - \delta'_{clf_i})\nabla_i W_{ij} \tag{34}$$

with

$$\delta'_{clf_j} = \begin{cases} \delta'_{clf_i} & \text{if j} \in \text{fluid} \\ 0 & \text{if j} \in \text{boundary} \end{cases} \tag{35}$$

and

$$\delta'_{clf_i} = \hat{v}_i \cdot \mathbf{n}_i. \tag{36}$$

2.7. Boundary conditions

The modified dynamic boundary conditions (mDBC) are used in this study due to its accurate estimation of density on boundary particle and the reduction of an unphysical gap between fluid and boundary particles in original DBC method [12]. In the mDBC method, a ghost particle is generated within the fluid for each boundary particle. The density and its gradient on the ghost particle are calculated by the linear interpolation over the neighbouring fluid particles. This linear interpolation is the first-order consistent interpolation proposed by Liu and Liu [24]. In the mDBC, the density ρ_i on boundary particle i is obtained by extrapolating to the boundary particle node,

$$\rho_i = \rho_g + (\mathbf{r}_i - \mathbf{r}_g) \cdot \begin{bmatrix} \partial_x \rho_g \\ \partial_y \rho_g \\ \partial_z \rho_g \end{bmatrix}, \tag{37}$$

where ρ_g and $\mathbf{r}_g$ are the density and position of the ghost particle.

To apply the no-slip boundary condition, the velocity on boundary particle with a non-moving boundary is given as:

$$\mathbf{u}_i = -\mathbf{u}_g, \tag{38}$$

where the $\mathbf{u}_g$ is the velocity of the ghost particle.

2.8. Particle shifting algorithm

As a particle-based and truly Lagrangian method, the SPH particles are free to move so the particles may clump together due to the small gradient of the kernel when the particles get close to each other. This particle clumping results in the irregular particle distribution and introduces the error and numerical instability in the simulation. To avoid the particle clumping, Lind et al. [23] developed a particle shifting technique for free-surface flows (PST) where the particles are shifted after the properties on particles being updated to create a more uniform particle distribution. This PST is based on Fick's law of diffusion where the particles are shifted from the areas of high concentration to that of low concentration, which gives,

$$\mathbf{J} = -\mathcal{D}'\nabla C, \tag{39}$$

where $\mathbf{J}$ denotes the diffusion flux, $\mathcal{D}'$ presents the diffusion coefficient and C is the concentration. The ∇C on the particle i can be found in SPH form as,

$$\nabla C_i = \sum_j V_j \nabla W_{ij}. \tag{40}$$

The flux $\mathbf{J}$ is proportional to the velocity of particles so that the shifting distance $\delta\mathbf{r}_s$ can be found as,

$$\delta\mathbf{r}_s = -\mathcal{D}\nabla C, \tag{41}$$

where $\mathcal{D}$ is a new diffusion coefficient that incorporates both the diffusion coefficient $\mathcal{D}\prime$ and a constant of proportionality. This coefficient $\mathcal{D}$ is determined by the von-Neumann stability criterion as,

$$dt \leq 0.5\frac{h^2}{\mathcal{D}}. \tag{42}$$

Table 1
Algorithm of the C-CSF model.

Algorithm: C-CSF Algorithm
1. Calculate the matrix on particles by $\mathbb{L}_i^{-1} = \sum_j (\mathbf{r}_j - \mathbf{r}_i) \otimes \nabla_i W_{ij} V_j$
2. Obtain the minimum eigenvalue of particle i λ_i
3. Determine the normal to the interface by
$\nabla\lambda_i = \sum_j (\lambda_j - \lambda_i)(\mathbb{L}_i \nabla W_{ij}) V_j$ and $\mathbf{n}_i = -\frac{\nabla\lambda}{\|\nabla\lambda\|}$
4. Correct the normal for fluid-boundary interaction based on
$\theta_i^* = \begin{cases} \theta_e + (\theta_i - \theta_e)(d_i/R_i)^2 & \text{if } d_i < R_i \\ \theta_i & \text{otherwise,} \end{cases}$
5. Obtain the curvature and surface tension force by
$\kappa_i = \sum_j s_{ij} (\mathbf{n}_j - \mathbf{n}_i) \cdot (\mathbb{L}_i \nabla W_{ij}) V_j$ and $\boldsymbol{F}_i^\sigma = -\sigma \kappa_i \boldsymbol{n}_i \delta_{s,i}$
6. Transfer the surface tension force to the function
InteractionForcesFluid' and *KerInteractionForcesFluid* in CPU and GPU code, respectively

Thus, based on Lind et al. [23], the particle shifting distance $\delta\mathbf{r}_s$ can be found as,

$$\delta\mathbf{r}_s = -Ah^2 \nabla C_i, \tag{43}$$

where A is a case-dependent coefficient. The maximum allowable shifting distance should be less than $0.2h$ to avoid the particle penetration into the wall particles.

In DualSPHysics, the default PST is adapted from Skillen et al. [41] where the velocity of the particle is introduced to the coefficient $\mathcal{D}$. The reason for using PST from Lind et al. [23] instead of the default one is that the interior particle might be shifted insufficiently due to its very low velocity such that the surface tension acts on the particle, resulting in clamping and numerical error.

2.9. Time stepping

The symplectic predictor-corrector method implemented in DualSPHysics [10] is used. The first step in this time stepping method is to predict the dependent variables at half time-step $t + dt/2$. Then, these variables are corrected using the predicted time derivatives at half time-step $t + dt/2$. Finally, the corrected variables are used to update the new value at next time-step $t + dt$.

Based on the Courant-Friedrichs-Lewy (CFL) condition, the variable time step is obtained by the following criteria. The first one depends on the body force,

$$\Delta t_f = C_f min_i(\sqrt{h/\boldsymbol{f}_i}). \tag{44}$$

In DualSPHysics, the time step constraint for artificial viscous dissipation is given by,

$$\Delta t_{vis} = C_{vis} \min_i \frac{h}{c_0 + \max_j \frac{|h\mathbf{u}_i \cdot \mathbf{r}_{ij}|}{|\mathbf{r}_{ij}|^2 + 0.01h^2}}. \tag{45}$$

Finally, the stability condition for surface tension is given by [47],

$$\Delta t_{st} = C_{st} min_i(\sqrt{\frac{\rho_i R_i^2}{2\pi\sigma|\kappa_i|}}), \tag{46}$$

where $C_f = 0.1$, $C_{vis} = 0.1$ and $C_{st} = 0.05$ are the Courant numbers used herein for each criterion. Thus, the variable time step is obtained by,

$$\Delta t = min(\Delta t_f, \Delta t_{vis}, \Delta t_{st}). \tag{47}$$

3. Implementation in DualSPHysics

As stated in the introduction, there are multiple challenges to implement the unified SPH model accelerated on a GPU. Specifically, this relates to the creation of the new C++ functions and templates in the CPU code, and new CUDA kernels launched from within the existing CUDA code, such that the implementation of the new SPH formulation inherits the advantages of the highly-optimised DualSPHysics code.

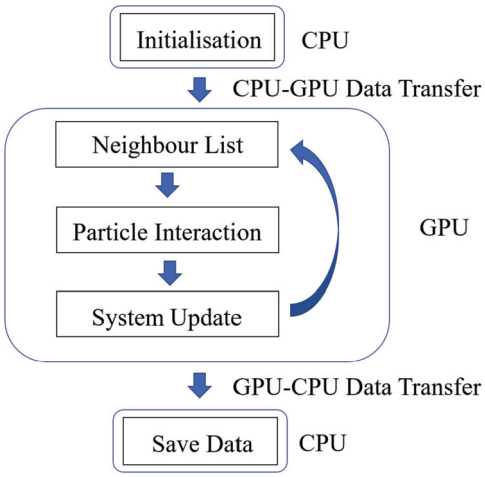


Fig. 2. Flow diagram of CPU-GPU key steps in DualSPHysics [10].

As shown in Fig. 2, the key steps in DualSPHysics code are: (1) creating the neighbour list for particle i, (2) computing the forces among particles, (3) updating the whole system. In this work, the main modifications of the code are in the second step as shown in Fig. 3.

In general, several options have been implemented to decide whether the surface tension model is activated. The constant parameters such as coefficient of surface tension are also applied as user-assigned variables. As described in Fig. 3, the main modifications in the DualSPHysics code are four functions to calculate the relevant variables including the correction matrix $\mathbb{L}$ and its minimum eigenvalue (Equation (15)), normal (Equation (13)), curvature (Equation (16)) and surface tension force (Equation (12)) before the force computation function *InteractionForcesFluid* in the CPU code and *KerInteractionForcesFluid* in the GPU code. After initialisation, the GPU-version of DualSPHysics performs the entire simulation on the GPU, only offloading data for outputting purposes.

The surface tension force algorithm calculation is shown in pseudo code of Table 1. To simplify the expression, four new CUDA kernels are named as Kernel-Matrix & MinimumEigenvalue (K-MEV), Kernel-Normal (K-N), Kernel-CorrectedNormal (K-CN) and Kernel-SurfaceTension (K-ST), respectively. The K-MEV estimates the correction matrix $\mathbb{L}$ and the corresponding minimum eigenvalue on each particle based on Equation (15). Two new arrays, one of *tmatrix3f* data type and one of *float* data type, are declared to store the matrix and the minimum eigenvalue. It is noted here that if the fluid-boundary interaction was considered, the matrix of boundary particles is also required so that the boundary particles within the kernel of fluid particle should have a full kernel. For example, as shown in Fig. 4, namely four layers of

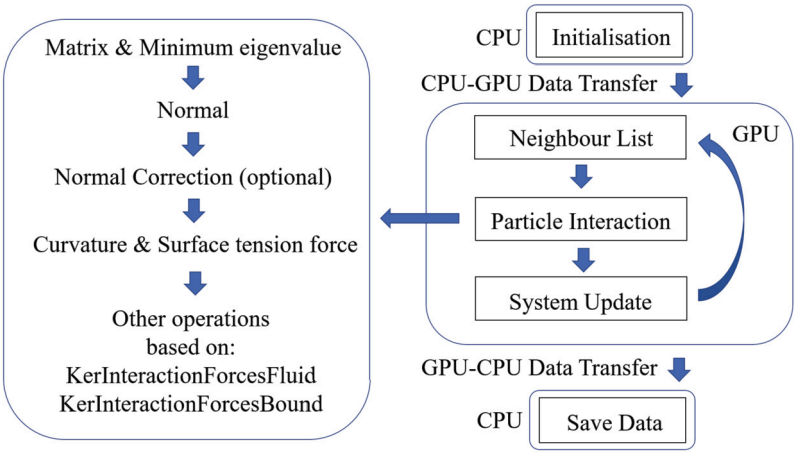


Fig. 3. Modifications for the implementation of the surface tension model in DualSPHysics.

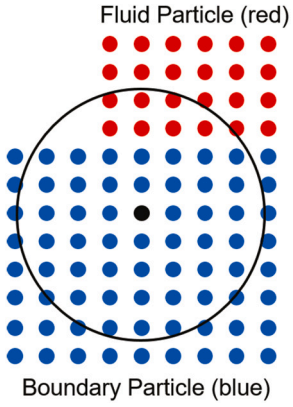


Fig. 4. Definition sketch of the support for the wall boundary particles, the black particle is the interpolating particle. Note, 8 layers of boundary particles are required to ensure accurate estimation of the eigen matrix and minimum eigenvalue.

boundary particles are needed for the case with the smoothing length $h = 2dp$, but here eight layers of boundary particles are needed for the surface tension model to ensure the accurate estimation of matrix and minimum eigenvalue. Therefore, the DualSPHysics option *cellfixed* should be activated in the .bat/.sh file to ensure the boundary particle has a full kernel because the original DualSPHysics only considers the boundary particles within the kernel of a fluid particle to reduce the computational cost. Another option for simplicity is to set the minimum eigenvalue of all the boundary particles to 1 which might result in errors as stated in Vergnaud et al. [47]. Considering the additional computation due to the extra layers of boundary particle required, the use of hardware acceleration is necessary.

The kernel K-N computes the normal vector on the fluid particles. It is noted that the extra layers of boundary particles are not required in this and the subsequent kernels. The gradient of the minimum eigenvalue is computed at first based on Equation (14). Then, the normal vector on each fluid particles can be obtained. The criterion of interfacial function g_i in Equation (24) and the unit wall normal $\mathbf{n}_{b,i}$ in Equation (26) are also found at this step.

Next, the normal obtained at the last step is corrected by K-CN if the fluid-boundary interaction was considered in the case. Then, the curvature and surface tension force are obtained by K-ST. Finally, the surface tension force is transferred to the force computation function *InteractionForcesFluid* in the CPU code and *KerInteractionForcesFluid* in the GPU code and added to the momentum equation.

The implementation of the new surface tension model has made no changes to the memory layout of the code.

4. Results and discussion

In this section, several test cases are conducted to validate the C-CSF model and CLF model in DualSPHysics. All the results are obtained based on GPU code. Following numerical tests, the results from the CPU and GPU versions of code are identical to within machine precision, thus CPU results will be omitted.

4.1. Drop deformation in 2-D and 3-D

Initially square (2-D) and a cubic (3-D) drops driven by the surface tension force are first investigated to verify the surface tension model. Three initial inter-particle distances are chosen as $dp = 0.02\ m$, $0.01\ m$ and $0.005\ m$, respectively, giving 441, 1681 and 6561 fluid particles for the 2-D case and 9261, 68921 and 531441 fluid particles for the 3-D case, respectively. The density of fluid is $\rho = 1\ kg/m^3$ and the dynamic viscosity is $\mu = 0.2\ Pa \cdot s$. The surface tension coefficient is set to $\sigma = 1\ N/m$. A square drop and a cubic drop with initial length $L = 0.4\ m$ are placed in the centre of the domains of the 2-D and 3-D cases, respectively. Based on the length L, the equilibrium radius of the final drop R_{eq} is given as,

$$R_{eq} = \begin{cases} L/\sqrt{\pi} & \text{in 2D} \\ (3/(4\pi))^{1/3} L & \text{in 3D.} \end{cases} \tag{48}$$

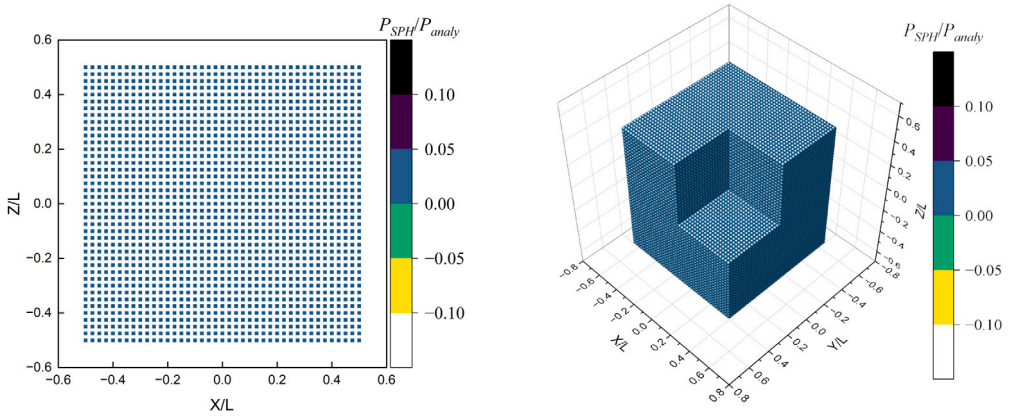
Then, the analytical pressure P_{analy} when the system reaches steady state is:

$$P_{analy} = \begin{cases} \sigma/R_{eq} & \text{in 2D} \\ 2\sigma/R_{eq} & \text{in 3D.} \end{cases} \tag{49}$$

In this validation, the equilibrium radii are $R_{eq} = 0.2257\ m$ and $R_{eq} = 0.2481\ m$ for the 2-D and 3-D cases, respectively. The analytical pressures are $P_{analy} = 4.43\ Pa$ and $P_{analy} = 8.06\ Pa$ for the 2-D and 3-D cases, respectively. The particle shifting algorithm is not used in this case due to the low Reynolds number so that the particle clumping is reduced by the viscous diffusion during the simulation.

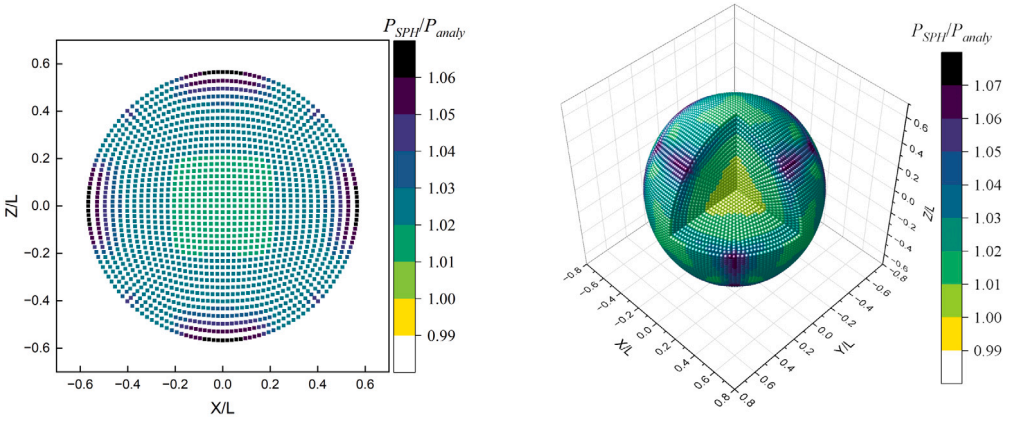
The particle position for both cases at $t = 0\ s$ and $t = 1\ s$ are shown in Fig. 5 and 6, respectively, where the term P_{SPH} is the pressure predicted by SPH.

The pressure profiles at $z = R_{eq}$ for the 2-D case and $y = z = R_{eq}$ for the 3-D case with three resolutions are shown in Fig. 7, where pressures are plotted against the non-dimensionalised distance from the centre of the drop $r^* = x/R_{eq}$. It can be seen that the pressure on free-surface particles for $dp = 0.005\ m$ is around 2-4% greater than the analytical solution in the 2-D case and is 1% less in the 3-D case. The pressure profiles agree well with the analytical solution and converge towards the analytical solution with increasing resolution.



(a) 2D case (b) 3D case

Fig. 5. Particle position at $t = 0\ s$, $dp = 0.01\ m$ for 2D and 3D drop deformation cases. One-eighth of the cube has been removed in 3D case to show the interior pressure distribution.



(a) 2D case (b) 3D case

Fig. 6. Particle position at $t = 1\ s$, $dp = 0.01\ m$ for 2D and 3D drop deformation cases. One-eighth of the sphere has been removed in 3D case to show the interior pressure distribution.

The convergence behaviours for 2D and 3D cases are shown in Fig. 8 where the L_2 error norm for pressure is defined as,

$$L_2 = \sqrt{\frac{\sum_i (P_{SPH} - P_{analy})^2}{N P_{analy}^2}}, \tag{50}$$

where the N is the number of particles and P_{analy} is the analytical pressure. The convergence rates are approximately 1.61 for the 2-D case and 1.54 for the 3-D case, which are typical values for an SPH scheme with no higher-order treatment [33,46].

4.2. Drop oscillation

The surface tension model is further validated by a single drop oscillation case. In this case, a drop with radius $R = 0.2\ m$ is generated. An initial velocity field $\mathbf{u} = (u_x, u_z)$ is applied to the drop [1],

$$\begin{aligned} u_x &= u_0 \frac{x}{r_0}(1 - \frac{z^2}{rr_0})\exp(\frac{-r}{r_0}) \\ u_z &= u_0 \frac{z}{r_0}(1 - \frac{x^2}{rr_0})\exp(\frac{-r}{r_0}), \end{aligned} \tag{51}$$

where r is the distance between the point (x, z) to the centre of the drop. The parameters $u_0 = 10\ m/s$ and $r_0 = 0.05\ m$ are chosen. The initial velocity field is shown in Fig. 9 where U is the velocity magnitude. The density of fluid is also set to $\rho = 1\ kg/m^3$. The dynamic viscosity is $\mu = 0.05\ Pa \cdot s$. The shifting algorithm from Lind et al. [23] is applied.

The snapshots of the motion at $t = 0.00s$, $0.16s$, 0.32 s and 0.48 s are shown in Fig. 10. The circular drop starts to oscillate under the effect of initial kinetic energy and tends to a stable circular drop with viscous dissipation. It should be noted that the final droplet drifts slightly due to the presence of parasitic velocities on the order of 0.005 m/s. The profiles of the mass centre of the upper right-square section of the drop in x and z direction with three resolutions are shown in the Fig. 11.

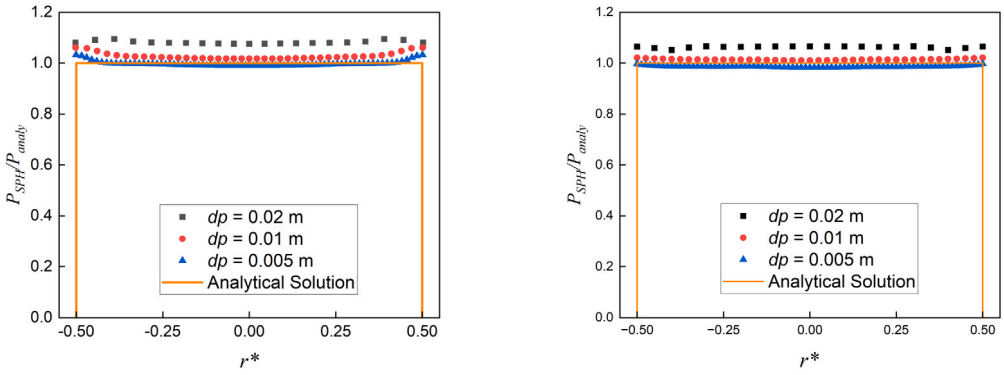
Next, the effect of the coefficient of surface tension on period time τ is investigated. The analytical period time is expressed as [1],

$$\tau = 2\pi \sqrt{\frac{R^3 \rho}{6\sigma}}. \tag{52}$$

Fig. 12 shows the computed period time with five coefficients of surface tension $\sigma = 0.4$, 0.6, 0.8, 1.0 and 1.2 N/m and three resolutions. For the smallest resolution $dp = 0.005\ m$, the largest deviation from the analytical solution is less than 2%.

4.3. Rayleigh-Plateau instability

The Rayleigh-Plateau instability is a common phenomenon in nature and industrial application. Specifically, the development of perturbation on the surface of a liquid ligament induces the break-up of the filament and formation of sub-droplets. In this section, a 2-D Rayleigh-Plateau instability case is conducted to validate the surface tension model.



(a) 2D case (b) 3D case

Fig. 7. Pressure profile at $z = R_{eq}$ for 2D and $y = z = R_{eq}$ for 3D drop deformation cases.

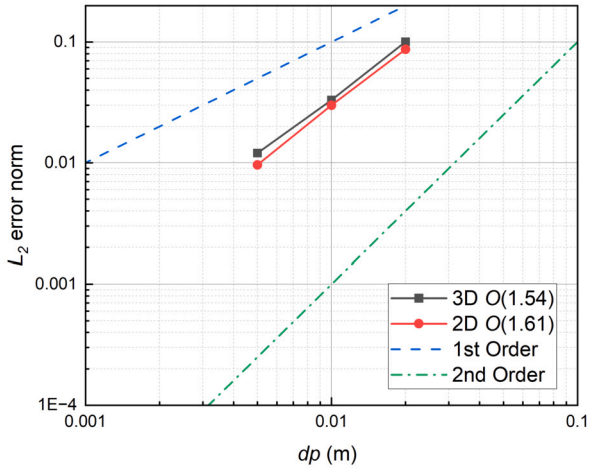


Fig. 8. Convergence behaviours for 2D and 3D drop deformation cases at $t = 1$ s.

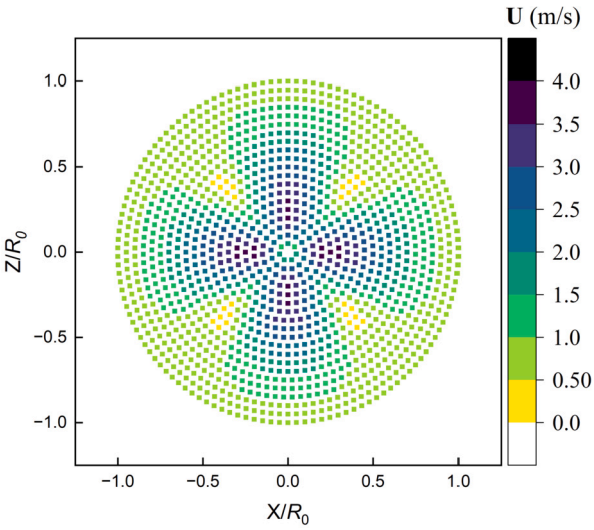


Fig. 9. Initial velocity field of the drop.

The numerical domain is a periodic and square domain with the length $L = 1\ m$. The fluid column is placed in the centre of the domain with the same length $L = 1\ m$ and radius $r_0 = 0.1L = 0.1\ m$. To simulate the initial perturbation on the column, and initial sinusoidal velocity field $\mathbf{u} = (u_x, u_z)$ is given as,

$$u_x = u_0 \sin\frac{2\pi x}{L}, \qquad u_z = 0, \tag{53}$$

where u_0 is the initial velocity and equals to $1\ m/s$.

From previous studies [13,36], the Weber number is $We = 1.4$ and the Reynolds number is $Re = 18$ in this case. The time is non-dimensionlised as,

$$t^* = \frac{t}{\sqrt{\frac{\rho r_0^2}{\sigma}}}. \tag{54}$$

Six resolutions are used which are $L/h = 64, 128, 192, 256, 384, 512$, giving 3328, 13312, 29568, 52736, 118272 and 209920 fluid particles, respectively and the smoothing length $h = 2dp$. The density of fluid is $\rho = 1000\ kg/m^{-3}$. Thus, based on the value of We and Re, the surface tension coefficient is $\sigma = 71.429\ N/m$ and the dynamic viscosity is $\mu = 5.556\ Pa \cdot s$. The particle shifting algorithm from Lind et al. [23] is used. The evolution for resolution $L/h = 512$ is shown in Fig. 13 and the surface tracking measure tool in DualSPHysics is used to capture the interface.

The growth rate profile $(r_{max} - r_0)/r_0$ is used to describe the behaviour of the fluid ligament. The results are compared with the data from Olejnik and Szewc [36] as shown in Fig. 14. It can be seen that the results of lower resolutions ($L/h = 64$ and $L/h = 128$) diverge from the results of finer resolution in the later stage which suggests a sufficiently fine resolution is required for this case. It is found that all the results agree well with the data from [36] in the early stage but a deviation could be observed in the late stage. To improve the results, the renormalized gradient from Bonet & Lok [39] and the optimized particle shifting scheme adapted from Khayyer et al. [21] have been applied. The improved results are shown in the Fig. 15. It can be seen that the deviation has been decreased but still exists. This deviation is likely be due to a multi-phase surface tension model being used in [36] as the fluid particles on the free surface have a truncated kernel support in the single-phase model.

4.4. Tests for the evolution of a square drop on a solid surface

To further validate the capability of surface tension model on fluid-boundary interaction, the case of an initially square drop on a solid boundary is conducted. As shown in Fig. 16, a fluid square is generated on a flat surface. Three initial inter-particle distances are used, which are $dp = 0.0001\ m, 0.00005\ m$ and $0.000025\ m$, respectively, giving 441, 1681 and 6561 fluid particles, respectively. The smoothing length is $h = 2dp$. The length of initial square drop is $L = 0.002\ m$. The initial radius of the equivalent semicircular drop can be determined noting the fluid square has the same area so that the equivalent initial radius of the semicircular drop is $R_{eq} = \sqrt{2L^2/\pi} \approx 0.001596\ m$. The properties

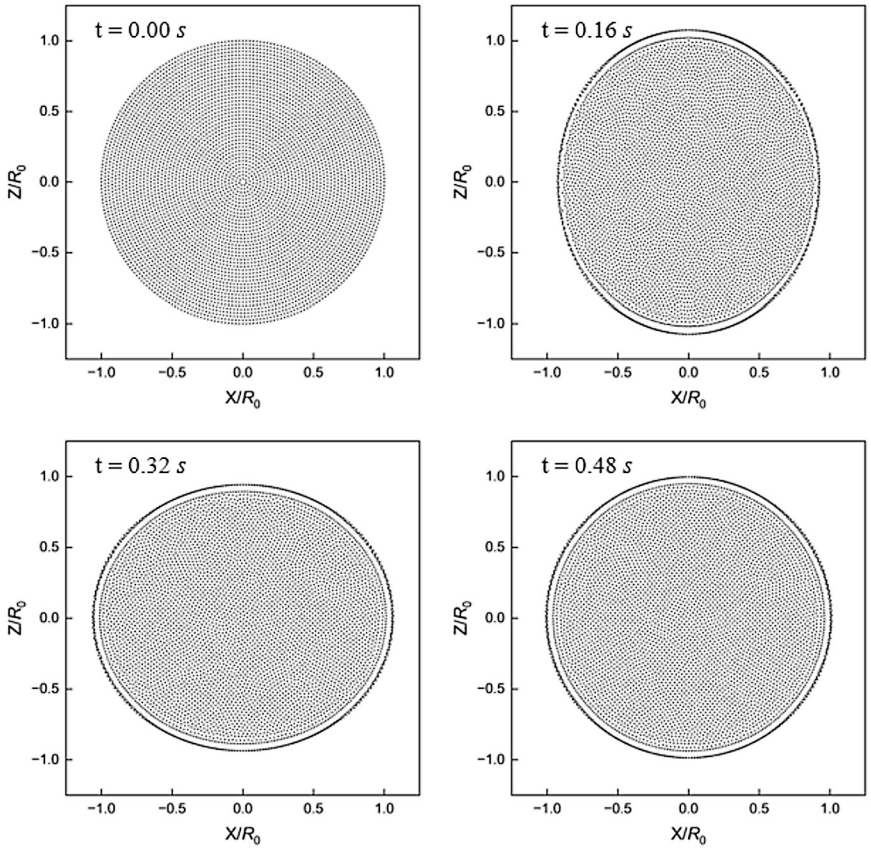


Fig. 10. Drop oscillation process at $t = 0.00$ s, 0.16 s, 0.32 s and 0.48 s.

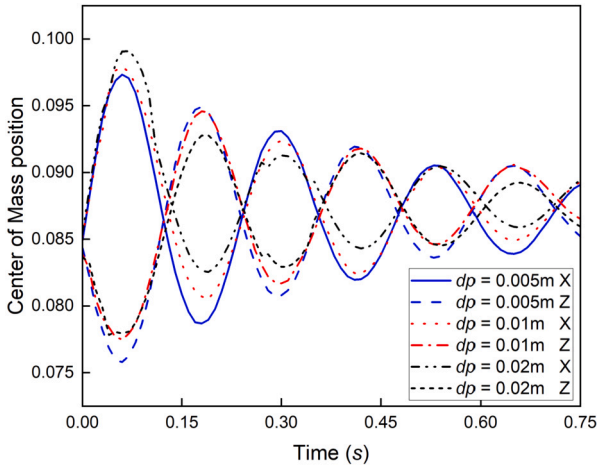


Fig. 11. The profile of centre of mass of the upper right-quarter section of the drop.

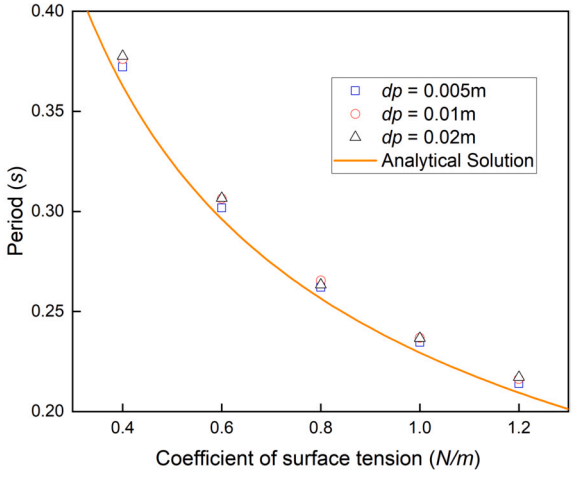


Fig. 12. Convergence of oscillation period.

of the fluid particles are: coefficient of surface tension $\sigma = 0.0728\ N/m$, dynamic viscosity $\mu = 0.1\ Pa \cdot s$ and reference density $\rho = 1000\ kg/m^3$. The gravity force is not active. Five equilibrium contact angles are chosen which are $\theta_{eq} = 30^o, 60^o, 90^o, 120^o$ and 150^o. The particle shifting algorithm from Lind et al. [23] is used.

The evolution of an square drop on a solid boundary with $\theta_{eq} = 90^o$ and $dp = 0.000025\ m$ is shown in Fig. 17. It can be seen that the surface tension model and the contact angle correction are effective. The distribution of the inner fluid particles is smooth due to the particle shifting algorithm. However, the cluster of the outermost two layers of particles could be observed and there is a gap between the free-surface particles and the inner particles. This gap can cause the outermost layer of the inner particles to be identified as free-surface particles and experience a surface tension force which is not physically correct. In addition, the cluster of the free-surface particles results in the pressure on boundary particles close to the corner of the drop being much greater than the analytical solution. Particles away from the fluid domain are also observed in this region especially for the smaller contact angle such as $\theta = 30^o$.

The kinetic energy profile for each contact angle during the simulation is shown in Fig. 18. It can be seen that the kinetic energy for each contact angle rises to its maximum value in the early stage due to the drop deformation driven by the surface tension force. Then, the kinetic energy decreases as the drop tends to equilibrium state. The kinetic energy does not tend to zero due to the intrinsic parasitic currents in the

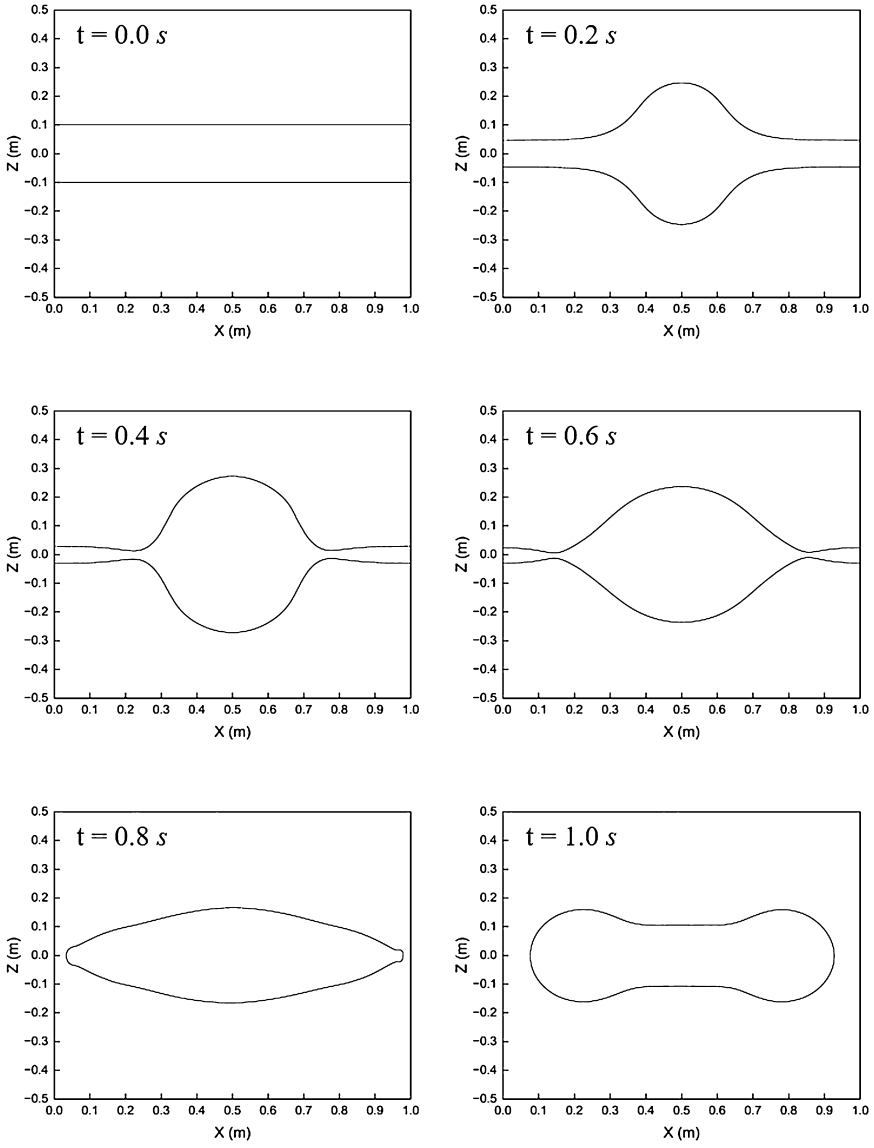


Fig. 13. Evolution of Rayleigh-Plateau instability, $L/h = 512$.

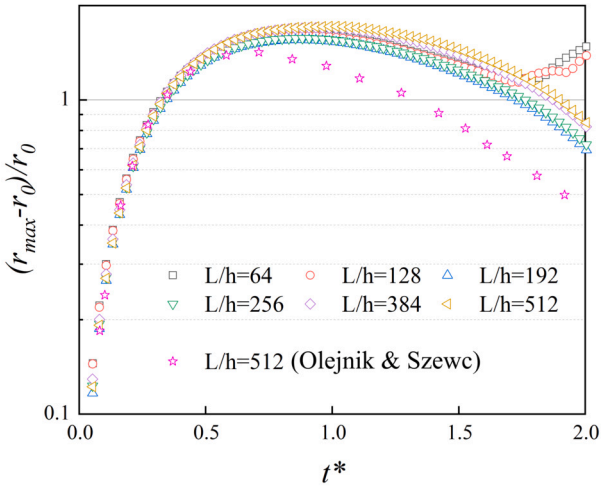


Fig. 14. The growth rate profile for Rayleigh-Plateau instability. The compared data is from Olejnik and Szewc [36].

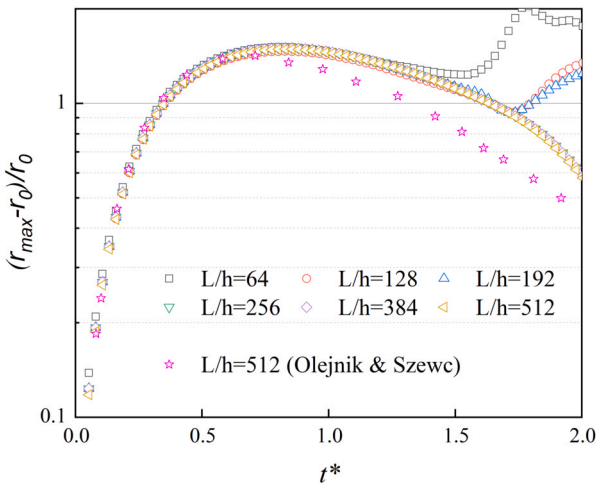


Fig. 15. The growth rate profile for Rayleigh-Plateau instability using renormalized kernel and optimised particle shifting scheme.

Table 2
Information on devices.

GPU	Quadro RTX 5000	CPU	Intel Xeon Gold 6130
Compute Capability	7.5	Clock Speed	2.10 GHz
Global Memory	16 GB	Cores	16
CUDA Cores	3072	Threads	32
Clock Rate	1.62 GHz	RAM	256 GB
Memory Bandwidth	448 GB/s	Memory Bandwidth	128 GB/s
TFLOPS (Single-precision)	11.2	\	\
ECC	YES	\	\

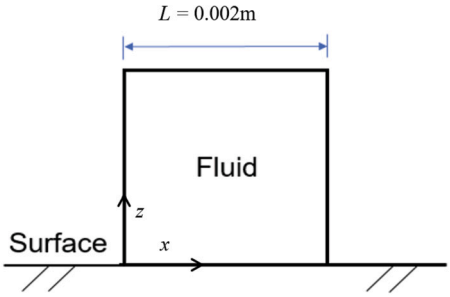


Fig. 16. Definition sketch: initial condition of drop contact angle case.

CSF model [15], which is a source of the error in the corner region. The cases with $\theta = 30^o$ and 150^o tend to have a relatively greater kinetic energy due to the error in the corner of the droplet.

The shape of the stable drop for cases with different contact angles is described as [35]:

$$\begin{aligned} R_{analy} &= R_{eq}\sqrt{\frac{\pi}{2(\theta_{eq} - \sin(\theta_{eq})\cos(\theta_{eq}))}}, \\ H &= R_{analy}(1 - \cos(\theta_{eq})), \\ WD &= 2R_{analy}\sin(\theta_{eq}), \end{aligned} \tag{55}$$

where R_{analy} is the analytical radius of the stable drop, H is the height of the drop and WD is the contact length between the drop.

The comparison of the height and width of the drop and the analytical solutions are shown in Fig. 19 and 20, respectively. From these two figures, a similar result could be observed as with the cases with $\theta = 30^o$ and 150^o which have a larger deviation. There are two main sources of the error. First, the error around the corner of the drop: it is difficult for SPH to capture this contact region with insufficient resolution particles. Second, the method to measure the height and width inherently contains errors. In this case, the free-surface measure tool in DualSPHysics is used where the measure error is in the scale of $O(h)$. Although the free surface is captured, it is difficult to determine the contact point in the corner of the drop which also contributes to the error.

A further comparison of C-CSF and CLF models on the measurement of the contact angle is also conducted. The contact angle is measured by [19]:

$$\theta_{DCA} = \frac{1}{\sum_j \delta_{clf_j}} \sum_j \theta_j \delta_{clf_j}, \tag{56}$$

where $\theta_j = \arccos(\hat{\mathbf{d}}_j \cdot \hat{\mathbf{n}}_j)$.

The absolute error under different equilibrium contact angles for different resolutions are shown in Fig. 21. Here, two different methods are applied: (1) C-CSF + CLF model shown in the solid lines (2) C-CSF model shown in the dashed lines. It can be seen that the C-CSF + CLF model generally has a smaller error for each resolution. This confirms that the additional contact line force model is required in the single-phase surface tension model of Vergnaud et al. to achieve closer agreement with the reference value.

5. Performance analysis

In this section, the GPU performance of the new CUDA kernels is validated by two test cases. The first case is a 3-D contact angle case to demonstrate the ability of the GPU to speed up and profile the new CUDA kernels. The second case is a 3-D dam-break case in the example of original DualSPHysics to estimate the extra computational expense and speed up ability of the modified code and the original DualSPHysics code. The details of each case are illustrated in the following discussion.

An NVidia Quadro RTX 5000 GPU and Intel Xeon Gold 6130 CPU are used to run GPU and CPU simulations, respectively. Both are designed for scientific computation and have been used to validate the hardware acceleration. The details of the devices are listed in Table 2. The maximum allowable number of particles in a single simulation case is around 36 million for the] modified single-phase GPU code.

The 3-D contact angle test case is used to investigate the speed-up of the GPU code. A cube with the initial length $L = 0.2\ m$ is placed on the centre of the boundary. The density, dynamics viscosity and the surface tension coefficient are $\rho = 1\ kg/m^3$, $\mu = 0.1\ Pa \cdot s$ and $\sigma = 1\ N/m$, respectively. Eight initial inter-particle distances, where the number particles ranges from 10000 to 6 million, are used to test the speed-up ability as shown in the Table 3. The value of speed-up is defined as the CPU time divided by the GPU time, $S = \frac{t_{CPU}}{t_{GPU}}$. In order to reduce the computation time, the physical time for each test resolution is set to 0.0001 s. The steady state 3D contact angle case is shown in Fig. 22 with approximately 270,000 fluid particles. It can be observed that a few particles (around three particles at each corner) are out of the fluid domain due to the sharp geometry.

In the 3-D dam-break test case, to assess the extra computational cost of the new surface tension computations, the surface tension coefficient is set as $\sigma = 0\ N/m$ to eliminate the surface tension force. The speed up of original code (without C-CSF model) based on 3-D dam-break test case is also conducted. The other setup is the same as the example 3-D dam-break case. The resolutions of the 3-D dam-break case are shown in the Table 4.

The run-time comparison between the CPU and GPU is shown in Fig. 23. The blue solid line presents the speed up of modified code based on the 3-D contact angle and the red dashed line is the original code based on the 3-D dam-break case. It is found that the speed-up increases significantly under 1 million particles. Then, it increases at a lower rate and further tends to a constant for the finer resolutions. It clearly shows the advantage of using the hardware acceleration offered by the GPU. The results also show that the ability of new CUDA kernels to speed up is stronger than that of original code. This might indicate that the CPU code for the surface tension model has room for improvement. Based on 3-D dam-break test case, the comparison of the run-time per physical second between the modified code (with the C-CSF model) and the original code (without the C-CSF model) is shown in the Fig. 24. The runtime comparison between the modified code and the original code exhibits a range of behaviours, spanning from 1.8 at the lowest resolution to 3.7 at the highest resolution. This trend shows that the computational performance of the single-phase surface tension GPU code progressively diminishes with increasing resolution.

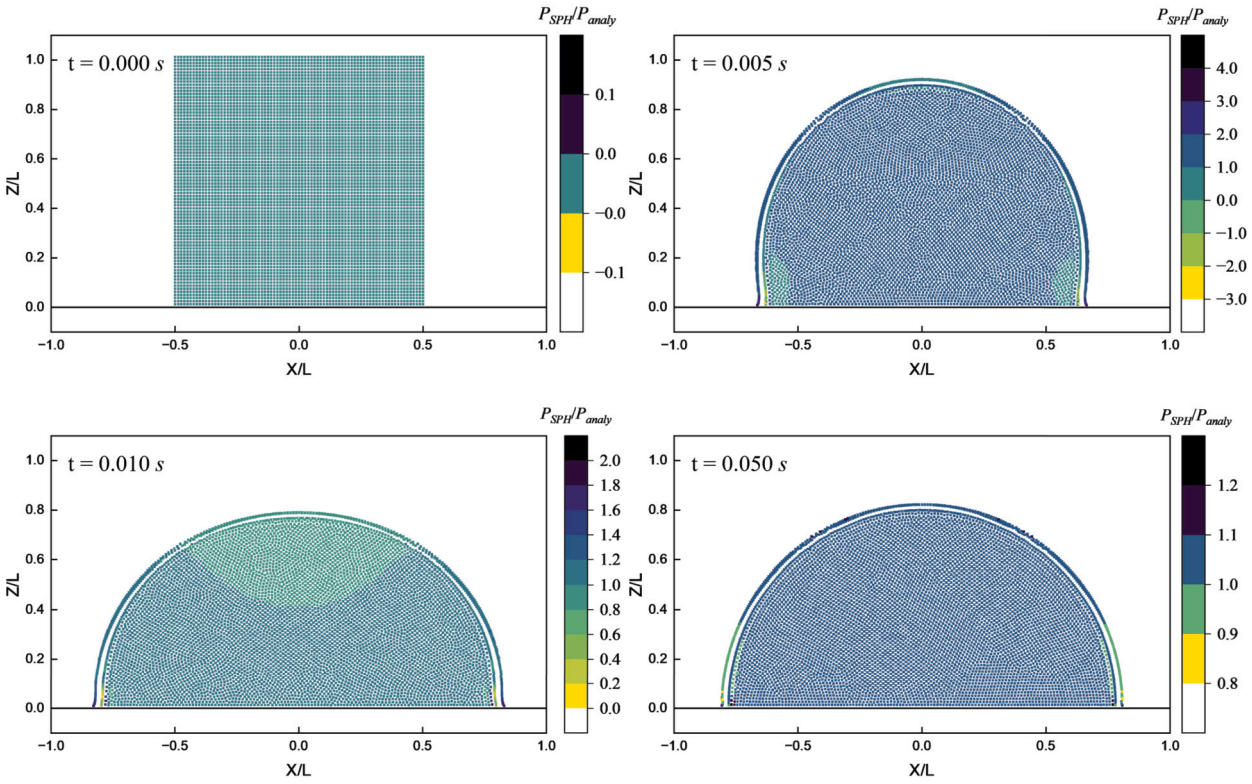


Fig. 17. Evolution of an square drop on a solid surface where the contact angle $\theta_{eq} = 90^o$ and resolution $dp = 0.000025\ m$.

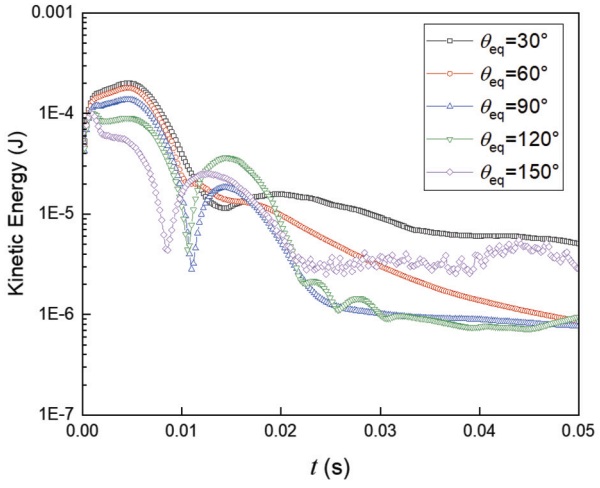


Fig. 18. Kinetic energy profiles for 5 contact angles during the simulation.

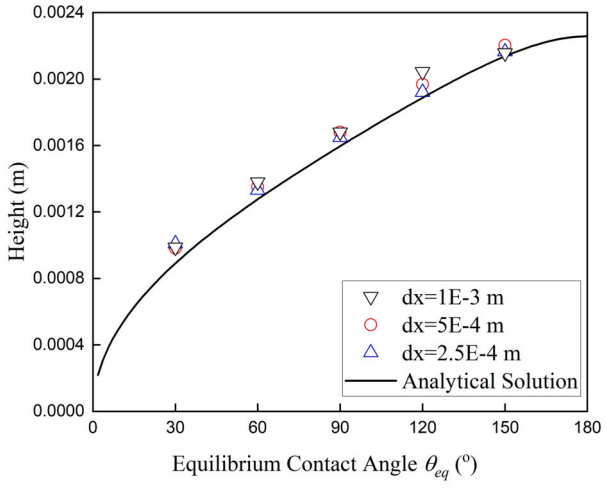


Fig. 19. Comparison of the height of drop with the analytical solution.

Table 3
The resolutions for the speed-up test case of the 3-D contact angle.

$dp(m)$	Fluid particles	Boundary particles	Total particles
0.02	1,331	3,844	5,175
0.01	9,261	14,884	24,145
0.005	68,921	58,564	127,485
0.004	132,651	91,204	223,855
0.0025	464,648	232,324	763,765
0.002	1,030,301	362,404	1,392,705
0.0016	2,000,376	565,504	2,565,880
0.00125	4,173,281	925,444	5,098,725

Table 4
The resolutions for the speed-up test case of the 3-D dam-break.

$dp(m)$	Fluid particles	Boundary particles	Total particles
0.04	9,360	26,143	35,503
0.02	79,380	126,392	205,772
0.01	652,212	489,324	1,141,536
0.008	1,290,096	760,348	2,050,444
0.0064	2,519,400	1,186,976	3,706,376
0.0056	3,760,011	1,532,218	5,292,229

The GPU profiling tool, NSight Compute, has been used to further investigate the specific performance metrics of the CUDA kernels in the GPU code, especially the new implemented kernels. The 3-D contact

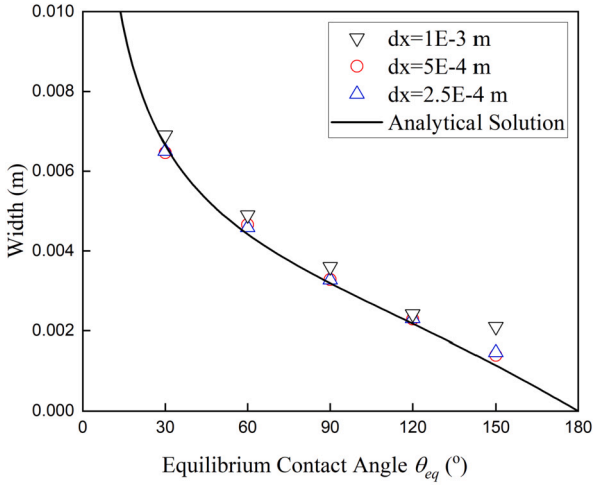


Fig. 20. Comparison of the width of drop with the analytical solution.

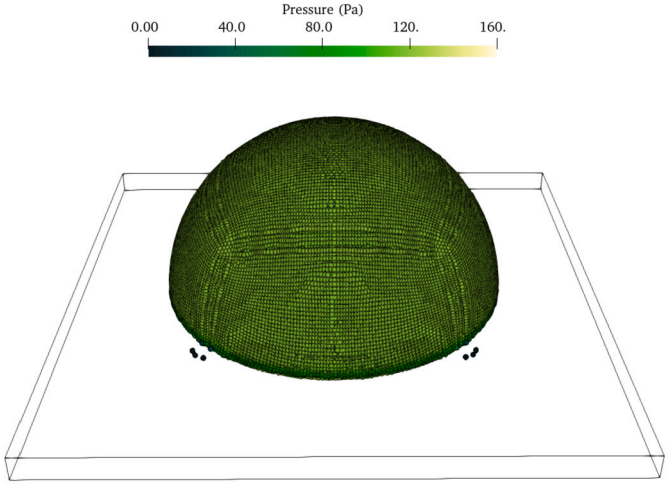
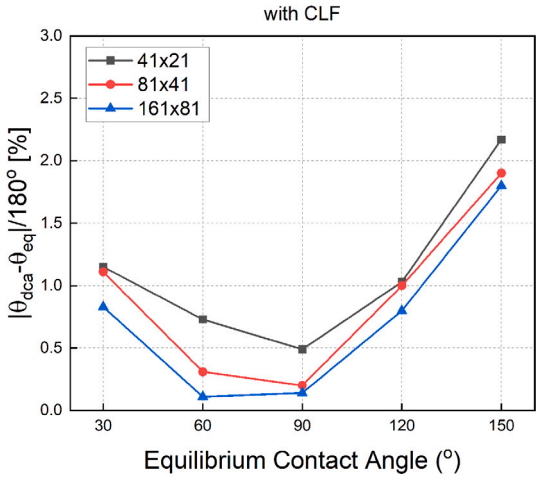
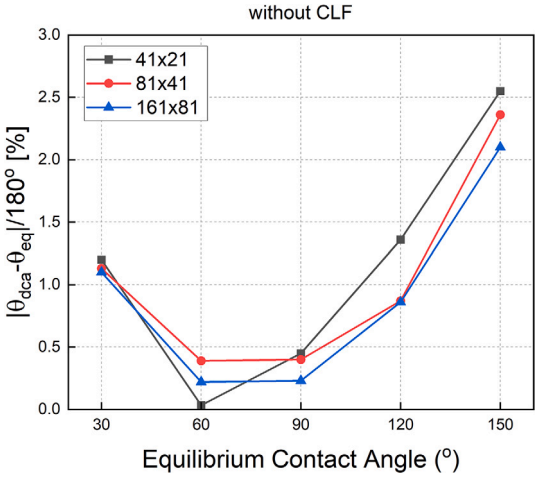


Fig. 22. A steady state particle distribution for 3-D contact angle case.



(a) with CLF



(b) without CLF

Fig. 21. Absolute error of the contact angle for different equilibrium contact angles (a) with CLF (b) without CLF.

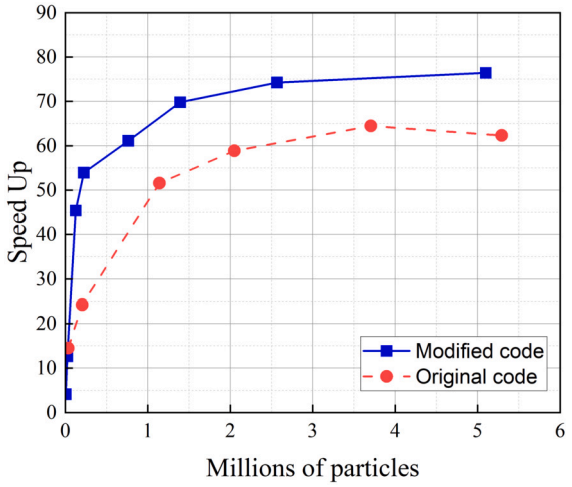


Fig. 23. Speed up between the CPU and GPU code for the 3-D contact angle test case. (For interpretation of the colours in the figure(s), the reader is referred to the web version of this article.)

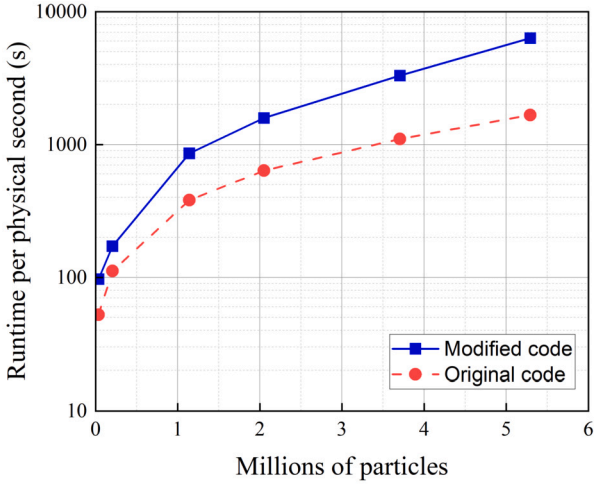


Fig. 24. Comparison of run-time per physical second between the modified code and original code for the 3-D dam-break test case.

Table 5
Profiling data for the new implemented kernels based on 3-D contact-angle case ($dp = 0.00125$ m) where the superscript $*$ denotes the result of the unmodified code.

CUDA Kernels	K-MEV	K-N	K-CN	K-ST	KerFluid/*	KerBound/*
Duration (ms)	42.23	53.50	0.06	54.16	70.69/(70.23*)	0.411/(0.403*)
Compute Throughput (%)	72.73	78.66	19.39	80.66	81.12/(81.19*)	61.55/(62.53*)
Memory Throughput (%)	33.77	33.44	87.10	40.02	25.9/(25.9*)	30.81/(31.26*)
L2 Hit Rate (%)	91.51	88.57	9.36	88.54	86.61/(86.99*)	68.18/(68.68*)
Achieved Occupancy (%)	96.72	96.44	78.39	95.51	95.47/(95.51*)	82.81/(83.03*)
Average Active Threads Per Warp	30.95	30.86	25.08	30.56	30.55/(30.12*)	26.5/(24.98*)
No Eligible (%)	27.92	27.16	89.05	19.28	29.84/(29.76*)	33.2372/(33.12*)
TFLOP/s (single-precision)	3.111	2.720	0.581	2.379	2.457/(2.473*)	2.384/(2.43*)
Registers Per Thread	52	64	28	62	66/(59*)	45/(45*)

Table 6
Profiling data for the new implemented kernels based on 3-D dam-break case ($dp = 0.0056$ m) where the superscript $*$ denotes the result of the unmodified code.

CUDA Kernels	K-MEV	K-N	K-CN	K-ST	KerFluid/*	KerBound/*
Duration (ms)	24.87	31.35	0.059	30.44	64.27/(63.92*)	1.45/(1.44*)
Compute Throughput (%)	72.33	78.07	48.76	81.94	80.94/(80.91*)	70.71/(70.57*)
Memory Throughput (%)	32.66	32.41	83.31	40.33	26.13/(26.12*)	32.45/(32.38*)
L2 Hit Rate (%)	88.94	84.96	1.18	84.06	86.02/(87.03*)	76.36/(76.39*)
Achieved Occupancy (%)	96.94	97.11	82.34	96.18	95.75/(95.77*)	90.83/(90.94*)
Average Active Threads Per Warp	31.02	31.08	26.35	30.78	29.29/(29.29*)	27.44/(23.28*)
No Eligible (%)	27.20	26.28	89.48	17.94	29.63/(29.62*)	23.28/(28.11*)
TFLOP/s (single-precision)	3.237	2.841	0.207	2.502	2.470/(2.482*)	2.53/(2.54*)
Registers Per Thread	52	64	28	62	66/(59*)	45/(45*)

angle case with $dp = 0.00125$ m (results shown in Table 5) and the 3-D dam-break case with $dp = 0.0056$ m (results shown in Table 6 are used and the kernels at the first time step are profiled. The CUDA kernel *KerInteractionForcesFluid* and *KerInteractionForcesBound* have been abbreviated as *KerFluid* and *KerBound*, respectively, and the superscript $*$ denotes the result of the unmodified code. The performance metrics on each main CUDA kernel shown in the both tables includes [34]: (1) *Duration*, refers to the length of time that a given kernel or function takes to execute on the GPU. (2) *Compute Throughput*, refers to the rate at which a kernel or function is able to perform computations on the GPU. (3) *Memory Throughput*, refers to the rate at which data can be transferred between the GPU's memory and the CPU's memory. (4) *L2 Hit Rate*, the percentage of memory access requests that find the required data in the Level 2 (L2) cache of a CPU (5) *Achieved Occupancy*, measures the percentage of hardware resources on the GPU that are being utilized by a kernel. (6) *Average Active Threads Per Warp*, measures the average number of threads that are active in a warp at any given time during the execution of a kernel. (7) *No Eligible*, indicates that no eligible device was found that can execute the CUDA kernel; (8) *TFLOP/s*, stands for trillion floating point operations per second; (9) *Registers Per Thread*, measures the amount of register memory allocated to each thread in a CUDA kernel.

It is shown that the new CUDA kernels including computation among fluid particles contribute to the time consumption. It is also found that each new CUDA kernel has near identical performance metrics compared to the *KerFluid* and *KerBound* except the K-CN, which is the normal correction. For K-CN, the performance metric 'TFLOP/s' is around 14-18 times lower than other kernels and the performance metric 'No Eligible' hits 90%. Although K-CN accounts for 0.1% of the time among these kernels, it is necessary to optimize the code for this part. It also found that the modified and unmodified codes have the near identical performance metrics on CUDA kernels *KerFluid* and *KerBound*.

6. Conclusion

In this work, an accurate and robust single-phase surface tension model has been developed in the open-source SPH code DualSPHysics. The single-phase surface tension model has been extended with the addition of a contact line force. Several test cases including drop deformation, drop oscillation, Plateau-Rayleigh instability and contact angle have been conducted. It is shown a good agreement between the results and the analytical solutions. The contact line force model provides a slightly more accurate estimation on the contact angle. The new CUDA kernels have been profiled to provide a more detail information on the run time. It is shown that the ability of the new kernels to speed-up is up to 80 compared to the CPU version based on the contact angle case and is larger than the original code whose speed-up is around 65-70 based on the 3-D dam-break case. In addition, each kernel has a near identical performance metrics compared to the main kernel in the original code except the kernel to correct the normal.

CRediT authorship contribution statement

Chunze Cen: Conceptualization, Investigation, Methodology, Software, Validation, Writing- original draft, Writing- review & editing; **Georgios Fourtakas/Steven Lind/Benedict Rogers**: Conceptualization, Methodology, Supervision, Writing- review & editing;

Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: Chunze Cen reports financial support was provided by China Scholarship Council.

Data availability

Data will be made available on request.

Acknowledgements

The first author would like to acknowledge the financial support provided by the China Scholarship Council and the University of Manchester joint scholarship fund.

References

[1] S. Adami, X. Hu, N.A. Adams, A new surface-tension formulation for multi-phase SPH using a reproducing divergence approximation, J. Comput. Phys. 229 (2010) 5011–5021.

[2] N. Akinci, G. Akinci, M. Teschner, Versatile surface tension and adhesion for SPH fluids, ACM Trans. Graph. (TOG) 32 (2013) 1–8.

[3] S.L. Anna, Droplets and bubbles in microfluidic devices, Annu. Rev. Fluid Mech. 48 (2016) 285–309.

[4] J.U. Brackbill, D.B. Kothe, C. Zemach, A continuum method for modeling surface tension, J. Comput. Phys. 100 (1992) 335–354.

[5] T. Breinlinger, P. Polfer, A. Hashibon, T. Kraft, Surface tension and wetting effects with smoothed particle hydrodynamics, J. Comput. Phys. 243 (2013) 14–27.

[6] C.E. Brennen, Cavitation and Bubble Dynamics, Cambridge University Press, 2014.

[7] A. Chesters, G. Hofman, Bubble coalescence in pure liquids, in: Mechanics and Physics of Bubbles in Liquids: Proceedings IUTAM Symposium, held in Pasadena, California, 15–19 June 1981, 1982, pp. 353–361.

[8] A.J. Crespo, et al., DualSPHysics: open-source parallel CFD solver based on Smoothed Particle Hydrodynamics (SPH), Comput. Phys. Commun. 187 (2015) 204–216.

[9] T. Dauch, et al., 3D predictions of the primary breakup of fuel in spray nozzles for aero engines, in: High Performance Computing in Science and Engineering'20: Transactions of the High Performance Computing Center, Stuttgart (HLRS), 2020, 2021, pp. 419–433.

[10] J.M. Domínguez, et al., DualSPHysics: from fluid dynamics to multiphysics problems, Computat. Part. Mech. (2021) 1–29.

[11] N.N. Ehigiamusoe, S. Maxutov, Y.C. Lee, Modeling surface tension of a two-dimensional droplet using smoothed particle hydrodynamics, Int. J. Numer. Methods Fluids 88 (2018) 334–346.

[12] A. English, et al., Modified dynamic boundary conditions (mDBC) for general-purpose smoothed particle hydrodynamics (SPH): application to tank sloshing, dam break and fish pass problems, Comput. Part. Mech. (2021) 1–15.

[13] S. Geara, et al., A new SPH density formulation for 3D free-surface flows, Comput. Fluids 232 (2022) 105193.

[14] R.A. Gingold, J.J. Monaghan, Smoothed particle hydrodynamics: theory and application to non-spherical stars, Mon. Not. R. Astron. Soc. 181 (1977) 375–389.

[15] D.J. Harvie, M. Davidson, M. Rudman, An analysis of parasitic current generation in volume of fluid simulations, Appl. Math. Model. 30 (2006) 1056–1066.

[16] G. Hetsroni, A. Mosyak, E. Pogrebnyak, L. Yarin, Fluid flow in micro-channels, Int. J. Heat Mass Transf. 48 (2005) 1982–1998.

[17] C.W. Hirt, B.D. Nichols, Volume of fluid (VOF) method for the dynamics of free boundaries, J. Comput. Phys. 39 (1981) 201–225.

[18] X.Y. Hu, N.A. Adams, A multi-phase SPH method for macroscopic and mesoscopic flows, J. Comput. Phys. 213 (2006) 844–861.

[19] M. Huber, et al., On the physically based modeling of surface tension and moving contact lines with dynamic contact angles on the continuum scale, J. Comput. Phys. 310 (2016) 459–477.

[20] H.P. Kavehpour, Coalescence of drops, Annu. Rev. Fluid Mech. 47 (2015) 245–268.

[21] A. Khayyer, H. Gotoh, Y. Shimizu, Comparative study on accuracy and conservation properties of two particle regularization schemes and proposal of an optimized particle shifting scheme in ISPH context, J. Comput. Phys. 332 (2017) 236–256.

[22] J. Kordilla, A.M. Tartakovsky, T. Geyer, A smoothed particle hydrodynamics model for droplet and film flow on smooth and rough fracture surfaces, Adv. Water Resour. 59 (2013) 1–14.

[23] S.J. Lind, R. Xu, P.K. Stansby, B.D. Rogers, Incompressible smoothed particle hydrodynamics for free-surface flows: a generalised diffusion-based algorithm for stability and validations for impulsive flows and propagating waves, J. Comput. Phys. 231 (2012) 1499–1523.

[24] M. Liu, G.-R. Liu, Restoring particle consistency in smoothed particle hydrodynamics, Appl. Numer. Math. 56 (2006) 19–36.

[25] L.B. Lucy, A numerical approach to the testing of the fission hypothesis, Astron. J. 82 (1977) 1013–1024.

[26] M. Luo, A. Khayyer, P. Lin, Particle methods in ocean and coastal engineering, Appl. Ocean Res. 114 (2021) 102734.

[27] F. Macia Lang, A. Souto Iglesias, M. Antuono, A. Colagrossi, Benefits of using a Wendland kernel for free-surface flows, 2011.

[28] G.D. Martin, S.D. Hoath, I.M. Hutchings, Inkjet printing-the physics of manipulating liquid jets and drops in, J. Phys. Conf. Ser. 105 (2008) 012001.

[29] A. Mayrhofer, B.D. Rogers, D. Violeau, M. Ferrand, Investigation of wall bounded flows using SPH and the unified semi-analytical wall boundary conditions, Comput. Phys. Commun. 184 (2013) 2515–2527.

[30] J.J. Monaghan, Smoothed particle hydrodynamics, Annu. Rev. Astron. Astrophys. 30 (1992) 543–574.

[31] J.P. Morris, Simulating surface tension with smoothed particle hydrodynamics, Int. J. Numer. Methods Fluids 33 (2000) 333–353.

[32] J.P. Morris, P.J. Fox, Y. Zhu, Modeling low Reynolds number incompressible flows using SPH, J. Comput. Phys. 136 (1997) 214–226.

[33] A.M. Nasar, et al., High-order consistent SPH with the pressure projection method in 2-D and 3-D, J. Comput. Phys. 444 (2021) 110563.

[34] NVIDIA, NVIDIA nsight compute documentation, https://docs.nvidia.com/nsight-compute/, 2021.

[35] M. Olejnik, J. Pozorski, A robust method for wetting phenomena within smoothed particle hydrodynamics, Flow Turbul. Combust. 104 (2020) 115–137.

[36] M. Olejnik, K. Szewc, Smoothed particle hydrodynamics modelling of the Rayleigh-Plateau instability, J. Theor. Appl. Mech. 56 (2018).

[37] M. Ordoubadi, M. Yaghoubi, F. Yeganehdoust, Surface tension simulation of free surface flows using smoothed particle hydrodynamics, Sci. Iran. 24 (2017) 2019–2033.

[38] S. Popinet, Numerical models of surface tension, Annu. Rev. Fluid Mech. 50 (2018) 49–75.

[39] P. Randles, L.D. Libersky, Smoothed particle hydrodynamics: some recent improvements and applications, Comput. Methods Appl. Mech. Eng. 139 (1996) 375–408.

[40] R. Reitz, S. Lin, Drop and spray formation from a liquid jet, Annu. Rev. Fluid Mech. 30 (1998) 85.

[41] A. Skillen, S. Lind, P.K. Stansby, B.D. Rogers, Incompressible smoothed particle hydrodynamics (SPH) with reduced temporal noise and generalised Fickian smoothing applied to body–water slam and efficient wave–body interaction, Comput. Methods Appl. Mech. Eng. 265 (2013) 163–173.

[42] W. Sun, L. Zhang, K. Liew, Fast detection of free surface and surface tension modelling via single-phase SPH, Appl. Math. Model. 100 (2021) 33–54.

[43] M. Sussman, P. Smereka, S. Osher, A level set approach for computing solutions to incompressible two-phase flow, J. Comput. Phys. 114 (1994) 146–159.

[44] A. Tartakovsky, P. Meakin, Modeling of surface tension and contact angles with smoothed particle hydrodynamics, Phys. Rev. E 72 (2005) 026301.

[45] A.M. Tartakovsky, A. Panchenko, Pairwise force smoothed particle hydrodynamics model for multiphase flow: surface tension and contact line dynamics, J. Comput. Phys. 305 (2016) 1119–1146.

[46] R. Vacondio, et al., Grand challenges for Smoothed Particle Hydrodynamics numerical schemes, Comput. Part. Mech. 8 (2021) 575–588.

[47] A. Vergnaud, G. Oger, D. Le Touzé, M. DeLeffe, L.C-C. SF Chiron, Accurate, robust and efficient surface tension and contact angle models for single-phase flows using SPH, Comput. Methods Appl. Mech. Eng. 389 (2022) 114292.

[48] D. Violeau, B.D. Rogers, Smoothed particle hydrodynamics (SPH) for free-surface flows: past, present and future, J. Hydraul. Res. 54 (2016) 1–26.

[49] Z.-B. Wang, et al., An overview of smoothed particle hydrodynamics for simulating multiphase flow, Appl. Math. Model. 40 (2016) 9625–9655.

[50] R.L. Webb, N. Kim, Enhanced Heat Transfer, Taylor and Francis, NY, 2005.

[51] H. Wendland, Piecewise polynomial, positive definite and compactly supported radial functions of minimal degree, Adv. Comput. Math. 4 (1995) 389–396.

[52] A.L. Yarin, et al., Drop impact dynamics: splashing, spreading, receding, bouncing, Annu. Rev. Fluid Mech. 38 (2006) 159–192.

[53] M. Zhang, H. Zhang, L. Zheng, Simulation of droplet spreading, splashing and solidification using smoothed particle hydrodynamics method, Int. J. Heat Mass Transf. 51 (2008) 3410–3419.

[54] C. Zöller, N. Adams, S. Adami, A partitioned continuous surface stress model for multiphase smoothed particle hydrodynamics, J. Comput. Phys. 472 (2023) 111716.