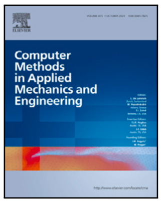
Computer Methods in Applied Mechanics and Engineering 429 (2024) 117199

Contents lists available at ScienceDirect

Comput. Methods Appl. Mech. Engrg.

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



Particle-based adaptive coupling of 3D and 2D fluid flow models

Pratik Suchde [*]

University of Luxembourg, Luxembourg
Fraunhofer ITWM, Germany

ARTICLE INFO

Keywords:
Model adaptivity
Discretization adaptivity
Meshfree
CFD
Shallow water
Free surface flow

ABSTRACT

This paper proposes the notion of model adaptivity for fluid flow modelling, where the underlying model (the governing equations) is adaptively changed in space and time. Specifically, this work introduces a hybrid and adaptive coupling of a 3D bulk fluid flow model with a 2D thin film flow model. As a result, this work extends the applicability of existing thin film flow models to complex scenarios where, for example, bulk flow develops into thin films after striking a surface. At each location in space and time, the proposed framework automatically decides whether a 3D model or a 2D model must be applied. Using a meshless approach for both 3D and 2D models, at each particle, the decision to apply a 2D or 3D model is based on the user-prescribed resolution and a local principal component analysis. When a particle needs to be changed from a 3D model to 2D, or vice versa, the discretization is changed, and all relevant data mapping is done on-the-fly. Appropriate two-way coupling conditions and mass conservation considerations between the 3D and 2D models are also developed. Numerical results show that this model adaptive framework shows higher flexibility and compares well against finely resolved 3D simulations. In an actual application scenario, a 3 factor speed up is obtained, while maintaining the accuracy of the solution.

1. Introduction

Flow phenomena that combines fluid dynamics on a surface and that in the bulk occur in many fields, ranging from spray coating to machine lubrication and tyre aquaplaning. To simulate such phenomena, various specialized thin film flow models have been developed that accurately capture surface-level flow effects. Popular thin film flow models include the shallow water equations [1] and the lubrication approximation [2], among many others [3–6]. All film flow models reduce the dimensionality of the governing equations, making them more efficient than bulk flow models for simulating fluids in a thin layer.

Such film models are typically applied to situations where a fluid film is the only fluid of interest, or where a fluid film forms an immiscible interface between two bulk fluids. However, these models have not been applied when there is a dynamic interplay between bulk and film flow. In a scenario where bulk flow develops into a thin film, the thin layer of fluid is usually treated just as any other bulk three-dimensional fluid region, using full 3D models. Simulations of such scenarios typically rely on significantly finer resolutions in the regions with thin film flow, which dramatically increases computation time. This makes full dynamic simulations unrealistic for actual applications. As a result, advances made in modelling thin fluid films have not been fully utilized in complex applications involving both surface and bulk flow.

In the present work, I propose a simulation framework for scenarios where bulk (3D) and thin film flow (2D) occur beside each other. The proposed algorithms can choose on-the-fly when and where a thin film flow model is applicable, and can shift the

* Correspondence to: Fraunhofer ITWM, Germany.
E-mail addresses: pratik.suchde@gmail.com, pratik.suchde@uni.lu.

https://doi.org/10.1016/j.cma.2024.117199
Received 24 May 2024; Received in revised form 25 June 2024; Accepted 25 June 2024
Available online 4 July 2024

governing equations automatically between a 3D flow model and a 2D model. Consider the example of a cleaning process in the food industry, where a large tray with food remnants is being cleaned with a jet of water (see Section 7.2 for details). The high velocity and flow rates of the jet would typically require a 3D flow model, like the Navier–Stokes equations, to be modelled accurately. As the jet hits the solid surface being cleaned, it would form a thin layer of fluid, which could be simulated quickly with a 2D thin film flow model. If sufficient fluid collects on the surface, the assumptions of a thin film model would no longer be applicable, and a 3D bulk flow model would be required again. Realizing this idea forms the main goal of the present work.

Coupling bulk and thin film flow has been widely done for modelling a tsunami landing on a shoreline (for example, [7–10]). Here, a thin film model, typically the shallow water equations, are used to model the flow of the ocean sufficiently far away from the coast, under the assumption that the ocean is significantly wider than deep in these regions. Near the coastline, the tsunami wave is modelled with a bulk flow model, typically the Navier–Stokes equations. The present work generalizes this existing literature by removing several key underlying assumptions present in tsunami modelling:

1. There is a fixed interface between the regions of the domain where the thin film model and the bulk flow model are applied.
2. This fixed interface can be prescribed a-priori, before starting the simulation.
3. The thin film flow is fully developed.

The present work removes these three assumptions by coupling 2D and 3D models without prescribing a fixed interface between them. The interface(s) between the two models will be automatically determined by newly developed algorithms, such that the location and presence of an interface can change in both space and time. This will allow moving interfaces, and even allow scenarios where the interface may completely disappear at an instant of time. This implies that it is possible for the entire simulation domain to be modelled by only a thin film model, or only a bulk flow model at a particular time step. This framework is referred to as being model adaptive, since the model (governing equations) being used are adaptively changed during the simulation.

The two models coupled in this work are (i) 3D bulk flow governed by the incompressible Navier–Stokes equations, discretized with an implicit meshfree collocation approach [11], and (ii) a pseudo-2D particle based thin film flow model called the discrete droplet method [12], with an explicit Lagrangian solution scheme that models film flow by tracking the movement of droplets on a surface. The framework introduced here can be easily adapted to be used with any Lagrangian meshfree or particle-based fluid solver and thin film solver. The key novelty and emphasis here is on the notion of model adaptivity, and not the choice of models or discretization methods.

Remark. The thin film flow model is referred to as *pseudo*-2D since it also covers flow over curved surfaces where a locally 2D model is used. For the sake of brevity, I drop the term pseudo, and refer to the thin film model as a 2D model.

It is important to note here that not only are the governing equations different in the two models, but the discretization itself is different as well. In the bulk flow approach, the entire flow domain is discretized. On the other hand, in the thin film model, only the surface is discretized, with the depth of the fluid film determined using an additional governing equation. Thus, in addition to being *model adaptive*, this framework is also *discretization adaptive*.

The paper is organized as follows. To start the paper, Section 2 introduces some necessary preliminaries without any novel work. This includes the terminology, and a brief summary of the bulk and thin film flow models used in the present work. Then, Section 3 presents an overview of the novel model adaptivity framework. The framework is split into three parts, which form the next three sections: Section 4 proposes how to determine where the underlying model needs to be changed, Section 5 presents how to change the model being used, and Section 6 presents how data can be communicated between two models that occur beside each other. Subsequently, numerical results, validations and applications are shown in Section 7 followed by a discussion on the limitations of the work in Section 8. The paper is then concluded with an outlook and summary in Section 9.

2. Preliminaries

In this section, I introduce both the bulk flow model, and the thin film model, and related preliminaries. I once again emphasize that the novelty in this work is not in the models themselves, but is in how the models are combined in an adaptive manner.

Throughout this work, I only consider domains in $\mathbb{R}^3$, with a three-dimensional bulk flow model, and a two-dimensional thin film model. However, for the ease of visualization, several schematics and figures shown in the methodology sections 3–6 illustrate domains in $\mathbb{R}^2$, with a two-dimensional bulk flow model and a one dimensional thin film model.

Remark. Since the 3D and 2D models will be solved one after the other, the model adaptivity and coupling presented in this work is independent of the spatial and temporal discretization of the individual models, under the restriction that both models use a meshfree/particle-based discretization.

2.1. Nomenclature

For the sake of clarity, this subsection defines the key terminology used.

- Throughout this work, the term *model* refers to the governing equations, with relevant initial and boundary conditions. Thus, the two models being used are a 3D bulk flow model, and a 2D thin film flow model.

- *Model adaptivity* refers to the process of adaptively choosing the model during a simulation. Note that this means a complete change of the governing PDEs, not just dropping one term from a PDE, as the term has been used in literature (for example, [13,14]).
- *Model transition* refers to the process of change or transition from one model to another. So either from the 3D model to the 2D one, or vice versa.

2.2. Meshfree terminology

The use of two meshfree methods, as opposed to mesh-based approaches, is crucial to the present work as it is straightforward to switch the underlying model being solved at a particle. Furthermore, meshfree methods also provide an easy approach for simulating free surface flow, which is central to the 2D-3D simulations considered here.

Consider a computational domain $\Omega = \Omega(t)$ partitioned into two non-overlapping regions, $\Omega(t) = \Omega^{3D}(t) \cup \Omega^{2D}(t)$ where Ω^{3D} is the subdomain where the 3D bulk flow model is being applied, and Ω^{2D} is the subdomain where the 2D thin film flow model is being applied. Both Ω^{3D} and Ω^{2D} could contain free boundaries. The computational domain is discretized with a cloud of $N^{\text{total}} = N^{2D} + N^{3D}$ points or particles, with $N^{2D} = N^{2D}(t)$ discretizing Ω^{2D}, and $N^{3D} = N^{3D}(t)$ discretizing Ω^{3D}. The terms point and particle are equivalent for the present work.

Remark. If the entire domain is governed with a bulk flow model at a particular instant of time t^*, we would have $N^{2D}(t^*) = 0$. Similarly, $N^{3D}(t^*) = 0$ holds if only the thin film flow model is being used at that time.

For both 3D and 2D domains, for a particle i, all approximations will be carried out on a compact support S_i of nearby particles, also referred to as its neighbourhood

$$S_i = \left\{ \vec{x}_j \,|\, \|\vec{x}_j - \vec{x}_i\| \le \frac{h_i + h_j}{2} \right\}, \tag{1}$$

where $h_i = h(\vec{x}_i)$ is the support radius or interaction radius at particle i. Note that $i \in S_i$. For more details on the support definitions, and other related meshfree basics, I refer the reader to [15] for 3D domains, and [12,16] for 2D domains and curved surfaces. The point cloud discretizing the initial domain is generated using a meshfree advancing front technique [17–19].

2.3. Bulk flow model

The bulk flow model used is the three-dimensional incompressible Navier–Stokes equations in a Lagrangian framework.

$$\nabla \cdot \vec{v} = 0, \tag{2}$$

$$\frac{D\vec{v}}{Dt} = \frac{\eta}{\rho}\Delta\vec{v} - \frac{1}{\rho}\nabla p + \vec{g}, \tag{3}$$

where $\vec{v}$ is the flow velocity, p is the pressure, ρ is the density, η is the dynamic viscosity, and $\frac{D}{Dt}$ is the material derivative. For the sake of simplicity, temperature and turbulent effects are not considered. Throughout this work, this bulk flow model is also referred to as the 3D model for short.

Temporal discretization

Time discretization is done using an implicit modified projection scheme [20]. The scheme starts with Lagrangian motion of particles [21]. This is followed by point cloud organization to maintain a quasi-regular point cloud and to prevent distortion [16,22]. Subsequently, an implicit intermediate velocity is determined, which is then projected to a divergence free space using an implicit pressure Poisson equation followed by the pressure update. For further details on the numerical scheme, I refer to [11,15,23].

Spatial discretization

The 3D domain is discretized using a meshfree method, as described in Section 2.2. The particles discretizing the domain can be of three types: (i) wall particles that lie on a solid wall, (ii) free surface particles, and (iii) interior particles that do not belong to any boundary.

The inter-particle distances are intrinsically linked to the interaction radius, as done in [15,24,25]. The minimum and maximum distance between two nearby particles, also referred to as separation and fill distance respectively, is fixed at $r_{\min}h$ and $r_{\max}h$ for interaction radius h and fixed parameters $r_{\min}$ and $r_{\max}$. In a moving Lagrangian framework, these distance criteria are enforced through the addition and deletion of particles. I refer to [16,22] for more details. Thus, h serves as both the interaction radius, and the resolution of the discretization defining the number of particles in the computational domain.

Derivatives are approximated using a meshfree collocation approach, referred to as the Generalized Finite Difference Method (GFDM) [26–28]. Unlike several other meshfree methods, the GFDM ensures discrete consistency up to the desired order of accuracy. This means that monomials up to the prescribed order of accuracy are differentiated exactly, even at the discrete level. For more details on GFDMs, I refer the reader to [29,30]. Throughout this work, an order of accuracy of 2 is prescribed.

2.4. Thin film flow model

There are two important considerations in the choice of the thin film flow model

1. The model should be able to capture the formation of thin fluid films, not just pre-existing films. This is essential for applications that consider bulk flow hitting a surface and forming a thin fluid film; see, for example, Section 7.2.
2. The model should allow free boundaries within the thin film flow.

Based on this, I choose the recently proposed discrete droplet method (DDM) [12,31] as the thin film flow model. The DDM is a meshfree approach that models incompressible flow in a thin fluid film using a Lagrangian framework by tracking the movement of individual fluid droplets. The momentum conservation equation is given by

$$\frac{D}{Dt}\vec{v}_{\text{film}} = -\frac{\eta}{\rho H^2_{\text{film}}}\vec{v}_{\text{film}} - \frac{1}{\rho}\nabla p + \vec{g}_{\parallel}\,, \tag{4}$$

where $\vec{v}_{\text{film}}$ is the velocity parallel to the surface on which the fluid is flowing, p is the pressure, $\vec{g}_{\parallel}$ is the component of the gravity parallel to the surface, ρ is the density, η is the dynamic viscosity, and H_{film} is the height or depth of the fluid film. The terms height and depth are used interchangeably in this work. The height of the fluid film is computed based on the aggregation of droplets. The height of the film at a particle i is given by

$$H_{\text{film},i} = \sum_{j\in S_i}\frac{\alpha}{\pi h_j^2}\exp\left(-\alpha\frac{\|\vec{x}_j-\vec{x}_i\|^2}{h_j^2}\right)V_j\,, \tag{5}$$

where h_j is the support radius of particle j, S_i is the support or neighbourhood of particle i (see Section 2.2), and V_j is the volume of particle j. Each particle is assumed to be a droplet of diameter d. Thus, we have $V_j = \frac{1}{6}\pi d_j^3$. The coefficient α is computed from a mass conservation condition, see [12]. The mass of a droplet particle i is given by $m_i = \rho_i V_i$. Note that both mass conservation and the divergence-free velocity condition are implicit to the model described here, see [12]. Throughout this work, this thin film flow model is also referred to as the 2D model for short.

Temporal discretization

An explicit temporal discretization is carried out for the thin film model. Time integration starts with the Lagrangian motion of droplets, followed by an explicit update of the momentum equation Eq. (4). Subsequently, the updated height function Eq. (5) is determined at the new droplet locations. For more details, I refer the reader to [12].

Spatial discretization

The domain discretization of the regions where the thin film model is being applied is done using particle or droplets, as explained above. A key point to note here is that only the surface is discretized, not the bulk. Droplets move on the surface, with the height of the film built up as described in Eq. (5).

Spatial derivatives are computed using a GFDM approach, just as the bulk flow model. I refer to [16,30], for details on computation of GFDM spatial derivatives on surfaces.

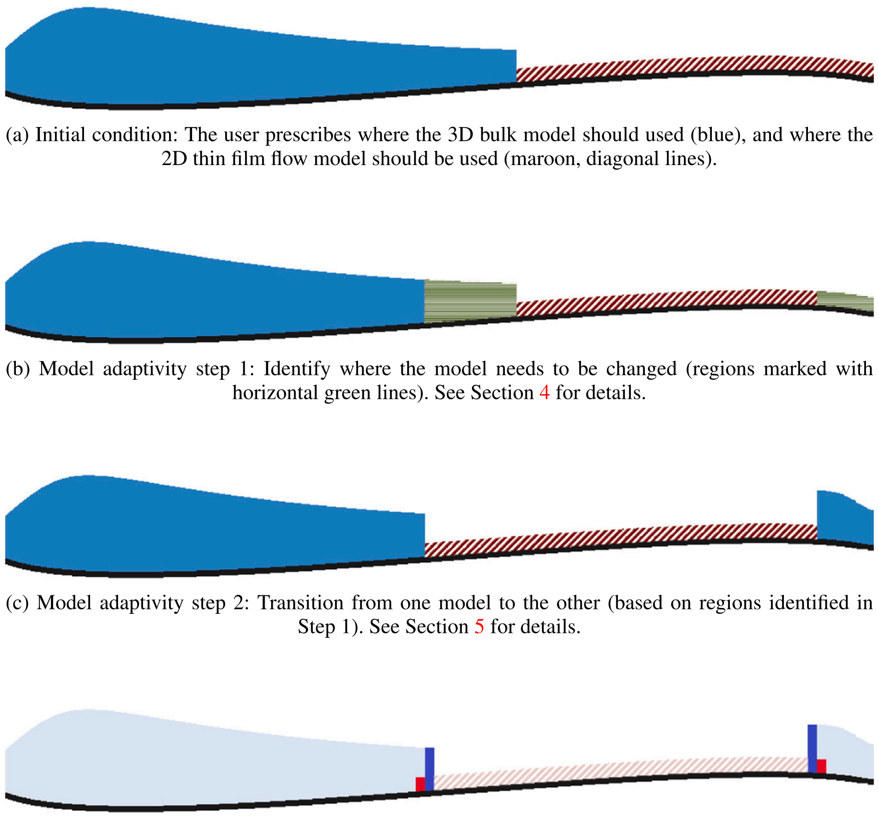
3. Model adaptivity overview

For the initial discretization and model selection, I assume that the user prescribes where in the domain the 3D and 2D models will be applied. Note that the user could also prescribe only one of the two models being present as the initial condition. Now consider the simulation time $t = t^*$, when either one or both models are present in the domain. The novel procedure to adaptively change the underlying model during a simulation is split into three questions

1. *Where* in the domain does the model need to be changed? This is answered in detail in Section 4.
2. In the parts of the domain where a desired model change is identified, *how* should the model transition be carried out? This question is tackled in Section 5.
3. Where both models occur beside each other, how to ensure *data communication* between the models? This issue is addressed in Section 6.

Remark. In existing literature coupling 3D and 2D flow that uses a fixed and pre-defined interface between the models, like that in tsunami modelling literature [7,32,33], only the third question is relevant. The first two questions are specific to the model adaptivity framework introduced here. Furthermore, for tsunami modelling, the solution to the third question relies on a pre-processing step before the simulation is started. In contrast, here, the data communication needs to be dynamic without any pre-processing possible, since the interface between models is not fixed.

Before each of these three questions are addressed individually in the coming three sections, I first present an overview of the numerical scheme in a schematic in Fig. 1 and in Algorithm 1.



(a) Initial condition: The user prescribes where the 3D bulk model should used (blue), and where the 2D thin film flow model should be used (maroon, diagonal lines).

(b) Model adaptivity step 1: Identify where the model needs to be changed (regions marked with horizontal green lines). See Section 4 for details.

(c) Model adaptivity step 2: Transition from one model to the other (based on regions identified in Step 1). See Section 5 for details.

(d) Model adaptivity step 3: Create handshake or buffer region for data communication between the two models (see Section 6 for details). The domain is shown with a high transparency. The buffer regions are shown in a brighter shade of the respective models: blue indicates the buffer region for the bulk models, and red indicates the buffer region for the thin film model.

Fig. 1. Schematic of model adaptivity between a 3D bulk flow model (blue) and 2D thin film flow model (maroon). The surface over which the fluid is flowing is shown in black. The thin film flow model is shown to be of a smaller height since it relies on a surface discretization only, while the entire bulk is discretized for the bulk flow model. Throughout this work, the bulk flow model is three-dimensional, while the thin film model is two-dimensional. Lower dimensional representations are used in this figure for ease of visualization. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

Algorithm 1 Model adaptivity overview: Numerical solution scheme

```
User-defined initial domain discretization
while Time-stepping loop do
    Check where the model needs to be changed                ▷ See Section 4
    if Model transition detected then
        Perform model transition                             ▷ See Section 5
        Delete buffer region from last time step             ▷ See Section 6
        Create new buffer region                             ▷ See Section 6
    end if
    Point cloud organization: addition, deletion of particles
    Bulk flow solver                                         ▷ See Section 2.3
    Thin film flow solver                                    ▷ See Section 2.4
    Post-processing calculations
end while
```

4. Detecting model transition

This section introduces the automatic domain decomposition into the subdomains Ω^{3D} and Ω^{2D} for the 3D and 2D models respectively. The initial definition of subdomains is user-prescribed. Subsequently, rather than performing a full domain decomposition at each time step, I only check for regions in the domain where the model being used needs to be changed from 2D to 3D, or vice versa. This event of changing the model is referred to as *model transition* throughout the present work. Before introducing the model transition criteria, I first establish a set of conditions that need to be fulfilled:

1. *Locality:* Any criterion to detect model transition must be local at each particle. For a particle i, the detection of model transition must be made based only on its neighbourhood S_i. This will avoid global checks, and make the detection more efficient.
2. *Resolution:* The choice of model should take into account the user-prescribed minimum (finest) resolution of the 3D model.
3. *Flexible:* Additional user-defined criteria for model selection should be allowed.

All particles are evaluated separately to determine if model transition is needed. Particles that satisfy all criteria presented below are flagged for transition. The actual transition of particles from one model to another, see Section 5, is only done after all particles are evaluated. I further split the detection of model transition into two parts:

1. Detecting when a 3D particle needs to be transitioned to a 2D particle, see Section 4.1.
2. Detecting when a 2D particle needs to be transitioned to a 3D particle, see Section 4.2.

4.1. Detecting 3D to 2D transition

Consider a 3D boundary particle i on a solid wall. Since the 2D thin film approximation is only applicable on and near walls, only wall particles need to be checked for transition to a 2D model. I now present a series of criteria, each of which need to be satisfied for a 3D particle to be flagged for model transition.

4.1.1. Resolution

The first criterion to be considered is based on condition 2 of resolution dependence. I assume a user-defined resolution for the 3D model, which could vary in both space and time. If the resolution at particle i is deemed insufficient to apply the 3D model, then the particle is flagged for transition to the 2D model. Conversely, if the resolution is deemed sufficient for the 3D model, then model transition is not done at this particle, and further criteria of model transition (see below) are not evaluated. This creates a condition between the resolution of the 3D particle and the height of the thin film model if it were to be applied at that location. Details on how the resolution criterion is evaluated in practice is presented in Appendix A.

4.1.2. PCA: eigenvalues

The fundamental assumption in all thin film models is that the fluid has a depth much smaller than the characteristic length scale in the other directions. This assumption is key in determining whether a thin film model is applicable at a particular location. Thus, the applicability of a thin film model could be investigated by looking at the local geometric structure of the point cloud.

I investigate the applicability of a thin film model through a local application of Principal Component Analysis (PCA), which has been widely used for linear structure approximation of scattered data. To perform local PCA on a particle i with neighbourhood S_i, consider the eigenvalues of the local covariance matrix defined as

$$\mathbf{P}_i = \sum_{j \in S_i} \left(\vec{x}_j - \vec{x}_c\right)\left(\vec{x}_j - \vec{x}_c\right)^T , \tag{6}$$

where $\vec{x}_j$ is a column vector prescribing the location of particle j, and $\vec{x}_c$ is the centroid of all the neighbouring particles

$$\vec{x}_c = \frac{1}{n(S_i)} \sum_{j \in S_i} \vec{x}_j , \tag{7}$$

with $n(S_i)$ being the number of particles in the neighbourhood S_i. I refer the reader to [34] for more details on PCA computation.

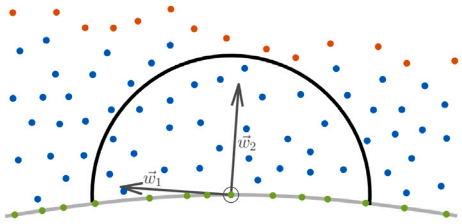
Remark. PCA has been successfully applied to prescribe a local coordinate system for point cloud surfaces and for surface normal computation (for example, [35,36]). For point cloud surfaces or manifolds, the relative sizes of the eigenvalues of the covariance matrix $\mathbf{P}_i$ reveals information about the (local) dimensionality of the manifold. For a surface in $\mathbb{R}^3$, one eigenvalue will necessarily be much smaller than the other two

$$\lambda_1 > \lambda_2 \gg \lambda_3 \tag{8}$$

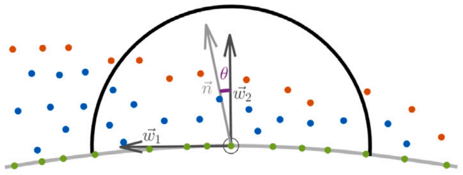
The eigenvectors corresponding to λ_1 and λ_2 will span the tangent space, while the eigenvector corresponding to λ_3 will be a surface normal vector.

Now, I extend this work on PCA for point cloud surfaces [35,36] to determine the applicability of a thin film model. This relies on the assumption that a bulk point cloud discretization of a thin film has similar structural properties to a point cloud discretizing a surface. While Eq. (8) may not hold, the relative sizes of the eigenvalues can be used to devise a measure to examine the local point cloud. Consider the eigenvalues $\lambda_1 > \lambda_2 > \lambda_3$. Heuristically, I propose a criterion that if λ_3 is "small enough", the model for particle i should be transitioned to a thin film approximation. Quantitatively, we need to define a threshold in the differences between λ_1, λ_2 and λ_3. For this, I consider the ratio

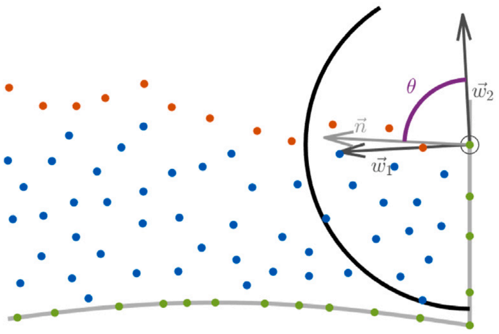
$$\lambda_3 = \epsilon_\lambda \frac{\lambda_1 + \lambda_2}{2} . \tag{9}$$



(a) None of the eigenvalues is significantly larger. No transition to the film model will take place.



(b) One eigenvalue (corresponding to eigenvector $\vec{w}_1$) is much larger than the other eigenvalue (corresponding to eigenvector $\vec{w}_2$). Furthermore, the eigendirection $\vec{w}_2$ is “close” to the normal $\vec{n}$. This particle will thus be transitioned to the thin film model.



(c) One eigenvalue (corresponding to eigenvector $\vec{w}_1$) is much larger than the other eigenvalue (corresponding to eigenvector $\vec{w}_2$). However, since the eigendirection $\vec{w}_2$ is “far” away from the normal $\vec{n}$, this particle will not be transitioned to the thin film model.

Fig. 2. PCA eigenvalue and eigenvector criteria for detecting bulk to thin film transition. Wall particles are marked in green, interior particles in blue, and free surface particles in orange. The wall particle at which the bulk to thin film transition criterion is being evaluated is marked with an additional black circle, with its neighbourhood highlighted, and normal $\vec{n}$ marked. The wall is marked in grey, and eigenvectors of the local variance matrix are marked as $\vec{w}_k$. Note that throughout this work a 3D bulk model and 2D thin film model is considered. This figure shows a lower dimensional schematic for the ease of visualization. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

For a point cloud discretizing a 2-manifold, we have $\epsilon_\lambda \ll 1$. The criteria of λ_3 being “small enough” for the applicability of a thin film model is realized by imposing a maximum value of ϵ_λ. If

$$\epsilon_\lambda < \epsilon_\lambda^{\max}, \tag{10}$$

I flag particle i for model transition. The value of $\epsilon_\lambda^{\max}$ is set by numerical trial and error to 0.15.

4.1.3. PCA: eigenvectors

In addition to the relative sizes of the eigenvalues, the direction of the corresponding eigenvectors also need to be examined. For a 3D particle i, let $\vec{w}_k$ for $k = 1, 2, 3$ be the normalized eigenvectors with unit length corresponding to the eigenvalues λ_k. Consider the scenario described above of a significantly smaller λ_3 compared to λ_1 and λ_2. Here, $\vec{w}_1$ and $\vec{w}_2$ form the principal components or directions of largest variation (of the particle locations) in the local neighbourhood of particle i. Furthermore, $\vec{w}_3$ is the orthogonal direction where the particle locations vary the least in the neighbourhood. For a change of model from 3D to 2D, we must ensure that the direction $\vec{w}_3$ corresponds to the direction of depth of the resultant thin film, see Fig. 2(b) . In certain geometric scenarios, it might be possible to have a significantly smaller λ_3 due to the boundary configuration. For example, consider a particle on a fixed wall right beside a free surface, as shown in Fig. 2(c). Here the small eigendirection points outside the domain, and does not indicate the presence of a thin fluid film.

This scenario is accounted for by imposing a condition on the direction of $\vec{w}_3$. Since the height of a fluid film at a location is, by definition, the extension of the film perpendicular to the surface, I check the direction of $\vec{w}_3$ in relation to the surface normal $\vec{n}$ at the particle. To account for a surface normal in either direction, I check the angular distance of $\vec{w}_3$ with both $\vec{n}_i$ and $-\vec{n}_i$. Thus, I impose the restriction of

$$|\vec{n}_i \cdot \vec{w}_3| < \cos(\theta^*)\,, \tag{11}$$

for a particle to be considered for transition to the thin film model. Here, θ^* is the threshold of maximum deviation allowed. Through numerical trial and error, I set a maximum deviation of the two vectors to be $\theta^* = \frac{\pi}{6}$.

In addition, to these criteria, I allow for user-defined criteria for model transition. An example of such a situation is shown with the numerical results in Section 7.3. The overall check of 3D to 2D model transition is summarized in Algorithm 2

Algorithm 2 Detecting model transition: 3D to 2D

```
Output: Flagged particles for bulk to thin film transition

function Loop(i: all 3D wall particles)
    TransitionFlag(i) ← false                          ▷ Not flagged for model transition
    if Resolution criterion then                       ▷ See Section 4.1.1
        if Eingenvalue and eigenvector criteria then   ▷ Eq. (10), Eq. (11)
            TransitionFlag(i) ← true
        end if
        if User-defined criteria then
            TransitionFlag(i) ← true
        end if
    end if
end function
```

4.2. Detecting 2D to 3D transition

Detecting model transition from the 2D thin film model to the 3D bulk flow models follows a similar, but simplified, procedure as the converse direction explained above in Section 4.1. Consider a 2D particle i. Note that by definition of the 2D model, all thin film particles are wall particles.

The primary criterion for this transition is the resolution. If the user-prescribed resolution of the *3D model* at the location $\vec{x}_i$ is deemed sufficient for the use of a 3D model, then the particle is flagged for model transition. This is quantified by comparing the 3D resolution with the computed film height at that location. Specifically, if the following condition holds, the particle i is flagged for model transition from 2D to 3D

$$H_{\text{film}}(\vec{x}_i) \geq h_{3\text{D}}(\vec{x}_i)\,. \tag{12}$$

Remark. At a given location $\vec{x}_0(t)$, the user can prescribe different resolutions for the thin-film and bulk flow models. $h_{3\text{D}}(\vec{x}_i)$ denotes the smoothing length and resolution (see Section 2.3) of the bulk model at the location of particle i, which might be different from the thin film resolution of particle i.

Consider the scenario introduced in Section 1 where fluid is collecting at a location $\vec{x}_j \in \Omega^{2D}$. The computed film height at that location $H_{\text{film}}(\vec{x}_j)$ keeps increasing as more fluid collects. Once Eq. (12) holds, the model applied at $\vec{x}_j$ will be changed to the 3D bulk flow model, as desired.

Furthermore, additional user-defined criterion are also allowed, and are showcased in Section 7.3.

4.3. Space and time smoothing

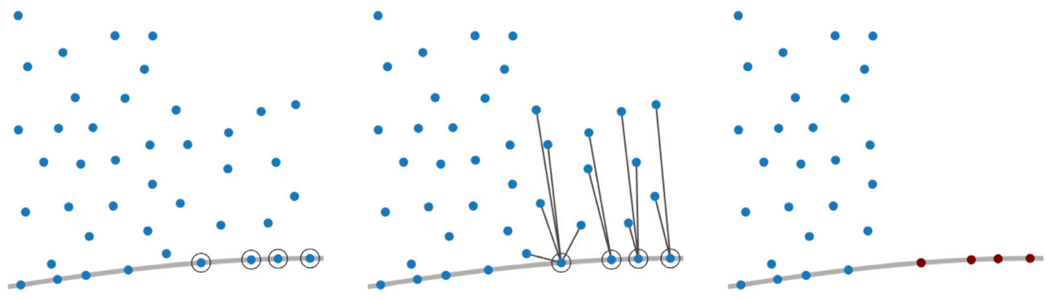
For model transition in either direction, in addition to the criteria defined above, I impose two further restrictions to prevent numerical instabilities.

1. If a single particle is flagged to undergo model transition, with no other particle in its neighbourhood being of the target model, model transition is not done. This prevents the scenario where a particle has no neighbours governed by the same model as itself.
2. Model transition is not performed for particles that have undergone a model transition in the previous time-step.

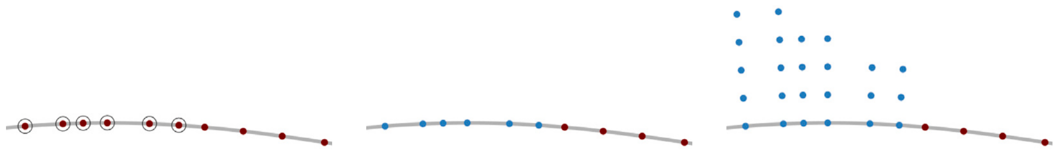
These restrictions impose a notion of smoothness on the model transition. Empirically, I observe that this helps maintain stability of the simulations.

5. Performing model transition

Section 4 dealt with detecting when a particle needs to be transitioned from one model to another. Once these checks are complete, all particles that need to undergo model transition are flagged. This section deals with how model transition is done for these flagged particles. There are two major challenges that need to be addressed to perform this transition:



(a) Transition from bulk particles to thin film particles.
Left: Wall particles identified for model transition highlighted.
Middle: Interior and free surface particles to be deleted identified and mapped to a particle undergoing model transition.
Right: Particles after model transition.



(b) Transition from thin film particles to bulk particles.
Left: Particles identified for model transition highlighted.
Middle: Flagged particles transitioned to the bulk model.
Right: New bulk particles added.

Fig. 3. Performing model transition from the bulk model to the thin film model (figure a), and from the thin film model to the bulk model (figure b). Bulk particles are shown in blue, and thin film particles in maroon. Note that throughout this work a 3D bulk model and 2D thin film model is considered. This figure shows a lower dimensional schematic for the ease of visualization. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

I *Discretization adaptivity:* As mentioned earlier, the 3D and 2D models have different types of discretization. In the 3D flow model, the entire domain is discretized with particles. In contrast, in the 2D thin film model, particles only discretize the surface over which the fluid flows. The presence of the fluid in the direction normal to the surface is represented by the height function, see Eq. (5). Thus, as the model is being transitioned from one to another, the type of discretization also needs to be changed.

II *Data transfer between models:* This second important challenge to address is how to accurately transfer system variables from one model to another during model transition.

Similar to Section 4, I address these issues separately for model transition from 3D to 2D in Section 5.1; and for 2D to 3D transition in Section 5.2. A schematic of the model transition in both directions is shown in Fig. 3.

Efficient neighbour searching is a critical part of every meshfree method. See, for example, [37–39]. Existing work towards this directly applies to both the models considered here. In this work, I maintain separate data structures (search trees) for the 3D model and the 2D model. Thus, a particle undergoing model transition must be removed from the data structure of the original model, and added to the data structure storing the new model. For MPI-based distributed memory parallel simulations, I observe that maintaining separate search trees resulted in better load balancing. However, this could be code dependent. Optimal strategy for load balancing remains a topic that needs investigation. Ideas in this regard from literature on the interaction of a fluid with a thin or flexible structure could be carried over to the present situation (see, for example, [40]).

5.1. 3D to 2D transition

Consider a 3D wall boundary particle i flagged for transition to the 2D model. Since the 2D thin film model only requires a discretization on the surface, a subset $\mathcal{T}_i \subset S_i$ of interior and free surface neighbours of i must be deleted. All system variables from the particles in $\mathcal{T}_i$ must be mapped to the updated 2D particle i. First, I deal with the question of how to determine $\mathcal{T}_i$, and then how data mapping will be done.

Determining $\mathcal{T}_i$

To ensure accurate data mapping, every particle being deleted needs to be mapped to a particle undergoing transition. For instance, for mass conservation, the mass of each particle being deleted must be lumped onto a nearby wall particle undergoing transition. For this, I defined $\mathcal{T}_i$ to be the set of all interior or free surface particles whose closest wall particle neighbour is i. The union of all these sets $\cup_i \mathcal{T}_i$ are the particles that have to be deleted once data mapping is complete. See the middle image of Fig. 3(a).

Computationally, this mapping is determined in a two-step procedure. First, for each 3D particle i flagged for model change to 2D, each of its interior and free surface neighbours are marked as candidates for deletion. Subsequently, consider a particle b marked as a candidate for deletion, and search the neighbourhood S_b for the nearest wall particle j^w. The superscript w indicates that the particle is a wall boundary particle. If j^w has been flagged for model transition, b is added to $\mathcal{T}_{j^w}$. If particle j^w has not been flagged for model transition, or if there is no wall particle in S_b, then particle b is removed from the list of candidate particles to be deleted.

Data mapping

Before the particles in $\cup_i \mathcal{T}_i$ are deleted, their data first needs to be mapped onto the particles undergoing model change. Most physical properties of the particle i in the 2D model are simply computed by an averaging of the data across the 3D model. For a system variable ϕ, its updated value after data model transition is determined as

$$\phi_i^{\text{new}} = \frac{\phi_i + \sum_{j\in\mathcal{T}_i} \phi_j}{1 + n(\mathcal{T}_i)}\,, \tag{13}$$

where $n(\mathcal{T}_i)$ is the number of particles in $\mathcal{T}_i$. This averaging of values is done for a majority of system variables: velocity $\vec{v}$, pressure p, density ρ, and viscosity η. Note that this averaging is also physically meaningful, since the 2D thin film flow model is obtained via a depth-averaging of the 3D bulk flow model [12]. However, for mass, rather than averaging, a mass lumping is needed to ensure mass conservation

$$m_i^{\text{new}} = \hat{m}_i + \sum_{j\in\mathcal{T}_i} \hat{m}_j\,. \tag{14}$$

Recall that for collocation based meshfree methods used for the bulk flow model, mass is not inherent to the numerical scheme. Rather, a representative notion of mass is prescribed for post-processing, which is used here. See [19] for details. The notation $\hat{m}$ denotes the representative mass for the bulk collocation solver, while the notation m denotes the actual mass for the mass-particle based thin film solver. The difference between these two methods are explained in detail in [19].

Once the updated mass m_i^{new} and density ρ_i^{new} are known, the droplet diameter can be determined with the relation $m = \rho V$, which gives $d_i^3 = \frac{6m_i^{\text{new}}}{\pi\rho_i^{\text{new}}}$. The height H_{film} of particle i in the 2D thin film model is set as the maximum normal distance of i from all particles in $\mathcal{T}_i$

$$H_{\text{film},i} = \max_{j\in\mathcal{T}_i} |(\vec{x}_j - \vec{x}_i)\cdot\vec{n}_i|\,, \tag{15}$$

where $\vec{n}_i$ is the unit boundary normal of particle i. Note that Eq. (15) only defines the height of the film immediately after model transition. During the thin film flow solver (see Algorithm 1), the height will be computed as per the procedure laid out in Section 2.4.

The procedure of transitioning a 3D particle to a 2D one is summarized in Algorithm 3, and illustrated in Fig. 3(a).

Algorithm 3 Performing model transition: 3D to 2D

```
Input: Flagged 3D particles that will undergo model transition

function LOOP(all 3D particles flagged for model transition to 2D)
    Mark all interior and free surface neighbours as candidates for deletion
end function

function LOOP($b$: all 3D candidates for deletion)
    find nearest wall boundary particle $j^w \in S_b$
    if $j^w$ is undergoing transition to 2D then
        Data mapping onto particle $j^w$
        Delete particle $b$
    else
        Remove $b$ from deletion candidate list          ▷ This particle will not be deleted
    end if
end function

function LOOP(all 3D particles flagged for model transition to 2D)
    Remove particle from data structure and neighbours lists of 3D model
    Add particle to data structure and neighbours lists of 2D model
end function
```

5.2. 2D to 3D transition

While the transition from 3D to 2D in Section 5.1 required the deletion of particles, the transition from 2D to 3D considered in this section requires the addition of new particles to create a bulk discretization. Consider a 2D particle i flagged for transition to 3D, with film height $H_{\text{film},i}$ and unit boundary normal $\vec{n}_i$ pointing inwards, i.e. in the direction of the fluid. For the change in discretization, I add particles along the direction $\vec{n}_i$ until a distance of $H_{\text{film},i}$ (see the rightmost image of Fig. 3(b)). This ensures that the resultant depth of the bulk model matches that of the thin film model.

Adding particles

Recall from Section 2.3 that the inter-particle spacing in the bulk model varies between $r_{\min}h$ and $r_{\max}h$ where $h = h(\vec{x}, t)$ is the *3D* resolution. To maintain these distances in the newly added particles, I assume 3D particles to be added at an approximate distance of

$$\tilde{\psi} = \frac{r_{\min}h + r_{\max}h}{2} . \tag{16}$$

As a result the number of particles added along $\vec{n}_i$ is given by

$$\gamma_i = \left\lceil \frac{H_{\text{film}_i}}{\tilde{\psi}_i} \right\rceil , \tag{17}$$

where $\lceil \cdot \rceil$ is the ceiling function. Thus, the actual distance between particles added is $\psi_i = \frac{H_{\text{film},i}}{\gamma_i}$, and the location of each new particle is given by $\vec{x}_{i_k} = \vec{x}_i + k\psi_i\vec{n}_i$ for $k = 1, 2, \dots, \gamma_i$. The particle furthest away from i, obtained by $k = \gamma_i$ is set as a free surface particle. Once particle addition is complete for all particles flagged for model transition, a free surface check is run on all newly added particles. This can follow any free surface detection method [41–44] in literature.

Data mapping

Data mapping for the newly created particles is straightforward. The velocity, pressure, density and viscosity for a newly created particle i_k is set to the same as that for particle i. The original mass of particle i is split evenly between all the newly created particles $\hat{m}_i^{\text{new}} = \hat{m}_{i_k} = \frac{m_i}{1+\gamma_i}$.

The procedure of transitioning a 2D particle to 3D one is summarized in Algorithm 4, and illustrated in Fig. 3(b).

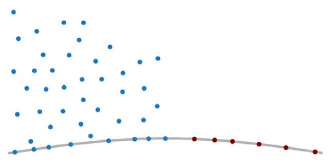
Algorithm 4 Performing model transition: 2D to 3D

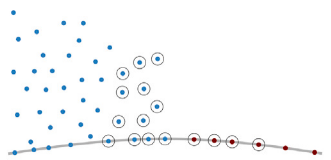
```
Input: Flagged 2D particles that will undergo model transition

function Loop(all 2D particles flagged for model transition to 3D)
    Data mapping for 3D model
    Remove particle from data structure and neighbours lists of 2D model
    Add particle to data structure and neighbours lists of 3D model
end function

function Loop(all particles just transitioned to 3D)
    Perform addition/deletion on the wall        ▷ to match desired resolution of 3D model
end function

function Loop(all particles just transitioned to 3D)
    Create new bulk particles in the normal direction
    Data mapping for new particles
    Set appropriate boundary type (interior/free surface) for new particles
end function
```

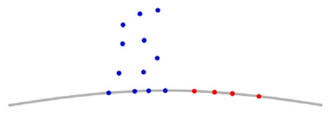
6. Data communication

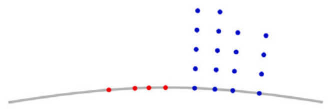
The last two sections dealt with where and how to perform model transition at each time step. Once all needed model change is performed, the final step in the model adaptivity framework is to perform time integration of the respective models. The models are solved sequentially, with the 3D solver running first. At a time step when particles of both models are present, there will be particles of one model with neighbours of another model, see Fig. 4(a). Since the two models have different discretization and different data stored, extra work needs to be done for data communication between the models.

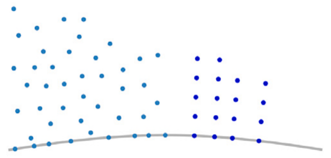
This same issue also occurs when two models are coupled without any adaptivity, as is done in existing literature. I adopt the usage of a so-called buffer zone from literature, also referred to as overlapping method [7,9], handshake region or blending region [45,46]. In this approach, in a small region centred at the interface between the models, a buffer zone exists where both models are solved. Data is communicated between models only through this buffer zone. In the present work, the buffer zone is populated with particles of each model. As a result, particles outside this buffer zone can be associated with neighbourhoods containing only particles of the same model (see Section 6.2), which significantly simplifies approximation procedures. Note that since the type of discretization is different for the 3D and 2D models considered here, the particles of the two models in the buffer region will have different locations.


(a) Bulk particles (blue, left) and thin film particles (maroon, right) beside each other. Ghost particles need to be created for data communication between these particles.


(b) Ghost particle sources: particles near particles of the other model are marked as sources for ghost particles.


(c) Intermediate ghost particles: Each source particle creates a ghost particle at the same location with the same model.


(d) Final ghost particles: After model transition on the set of intermediate ghost particles.


(e) All bulk particles: regular bulk particles (left), and ghost bulk particles (right, bright blue).

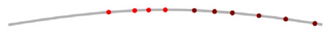

(f) All thin film particles: regular thin film particles (right), and ghost thin film particles (left, bright red).

Fig. 4. Creation of ghost particles. Bulk particles are shown in blue, and thin film particles in maroon. Ghost particles are shown in a brighter shade for each model. Note that throughout this work a 3D bulk model and 2D thin film model is considered. This figure shows a lower dimensional schematic for the ease of visualization. A representation of the same in $\mathbb{R}^3$ is shown in the results section in Fig. 5. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

Remark. The interface between two models is not stored explicitly nor implicitly through a level set or similar approach. In fact, the interface is not even computed. The term is used for a purely virtual interface across which a particle of a particular model interacts with a particle of the other model.

6.1. *Creating ghost particles*

The extra particles added in the buffer region are referred to as ghost particles, while non-ghost particles will be referred to as regular particles. The procedure of creating ghost particles and the buffer zone is presented in Algorithm 5, and visualized in Fig. 4. Details on how this procedure is realized in practice are presented in Appendix B. Note that since ghost particles are created for both the 2D and 3D models, this approach forms a two-way coupling of the models.

Algorithm 5 Creating ghost particles

```
function Loop($i$: all particles)                                  ▷ across both models
    Flag all other model particles within a distance of $h$        ▷ see Fig. 4(b)
end function

function Loop($j$: all flagged particles)                          ▷ across both models
    Create ghost particle of the same model as $j$ at location $\vec{x}_j$   ▷ see Fig. 4(c)
end function

function Loop(all newly created ghost particles)
    Perform model transition                                       ▷ according to algorithms 3 and 4
                                                                   ▷ see Fig. 4(d)
end function
```

In typical model coupling in existing literature, two models are solved beside each other and the interface is either fixed or can be tracked. In contrast, here, the interface between models is completely dynamic. Thus, the buffer region possibly needs to be updated at every time step. For a general scenario, where the interface is dynamic, I delete old ghost particles at the start of the time step, and then recreate new ghost particles after the model transition is complete. For the special case where the interface is fixed, ghost particles from the previous time step are not deleted and only their data is updated every time step.

The size of the buffer region is often considered a parameter in methods that couple two models across a fixed interface. In the present work, this is fixed to the interaction radius h at that location. Since all computations are local, based on neighbouring particles within the interaction radius, increasing the size of the buffer region beyond h will not increase accuracy.

6.2. Derivatives, approximations and time integration

Consider a particle i, with a set of neighbouring particles S_i. This set of neighbouring particles S_i only consists of neighbours being governed by the same model as i, and does not include ghost particles. Furthermore, consider the set S_i^g of ghost particle neighbours of i. Once again, S_i^g only includes ghost particles of the same model as i. The set S_i^g will be empty if particle i is sufficiently away from all particles of the other model. Let the complete neighbouring set, consisting of both regular and ghost neighbours, be represented as $\overline{S}_i = S_i \cup S_i^g$.

Remark. The free surface check for newly added particles in Section 5.2 must include ghost particle neighbours as well.

Before addressing derivative computation in the presence of ghost particles, I first recall the regular computation of derivatives in a GFDM approach without ghost particles. Consider a discrete function u defined at each particle location. The derivatives of u at particle i are approximated as

$$\partial^* u(\vec{x}_i) \approx \partial_i^* u = \sum_{j \in S_i} c_{ij}^* u_j \, , \tag{18}$$

where $^* = x, y, z, \Delta$ represents the differential operator being approximated, ∂^* represents the continuous *-derivative, and ∂_i^* represents the discrete derivative at particle i. The stencil coefficients c_{ij}^* are found either using Taylor expansions, or equivalently by ensuring that monomials up to the desired order are exactly differentiated [11,29].

In the presence of ghost particles, if $S_i^g \neq \emptyset$, all derivatives and approximations are computed using the extended neighbourhood $\overline{S}_i$, including both regular and ghost neighbours. Thus, we have

$$\partial_i^* u = \sum_{j \in \overline{S}_i} c_{ij}^* u_j \, , \tag{19}$$

$$= \sum_{j \in S_i} c_{ij}^* u_j + \sum_{j \in S_i^g} c_{ij}^* u_j \, . \tag{20}$$

Now, consider integrating from time level $t^{(n)}$ to $t^{(n+1)}$, where the bracketed superscript denotes the time level. Explicit time integration is straight-forward, by replacing u_j in Eq. (20) by $u_j^{(n)}$.

As explained above, due to the dynamic nature of the buffer region, the ghost particles are recreated at every time step. As a result, there is no need to update the system variables at the ghost particles. Thus. derivatives are never computed at ghost particles; and all large sparse implicit systems only contain regular particles, and no ghost particles. In the special cases when the same buffer region exists for multiple time steps, rather than recreating the ghost particles, the data at the ghost particles are recomputed as explained in Section 6.1. Since the data at ghost particles are not updated, derivative computation during implicit time integration is done using the unknown new time step value for regular neighbours, with the current time step value for ghost neighbours

$$\partial_i^* u = \sum_{j \in S_i} c_{ij}^* u_j^{(n+1)} + \sum_{j \in S_i^g} c_{ij}^* u_j^{(n)} \, . \tag{21}$$

The implicit terms in the first summation will be a part of the sparse system matrix, while the explicit terms of the second summation will be a part of the right-hand side.

7. Numerical results

The bulk flow and thin film models used here, and their implementations, have been introduced, verified and validated in the author's earlier work: see [12,31] for the thin film model, and [11,15,24,47] for the bulk flow model. Furthermore, the novelty in the present work lies in the adaptive combination of the models, and not the individual models themselves. Thus, this results section only focuses on the combination of the models, without any results with either of the models individually.

I consider test cases of increasing complexity. The simplest case tests the basic idea of model transition. Since both PDE solvers are switched off for this test case, it is presented in Appendix C. The first test case in this section tests various parts of the introduced framework, while the second case considers the full scheme. The last test case in this section considers the full scheme with added user-defined criteria for choosing the regions of applicability of the different models. A quick summary of the all the numerical simulations considered, and the motivation behind them, is described in Table 1. Note that all test cases are in $\mathbb{R}^3$ with a 3D bulk flow model and a 2D thin film model.

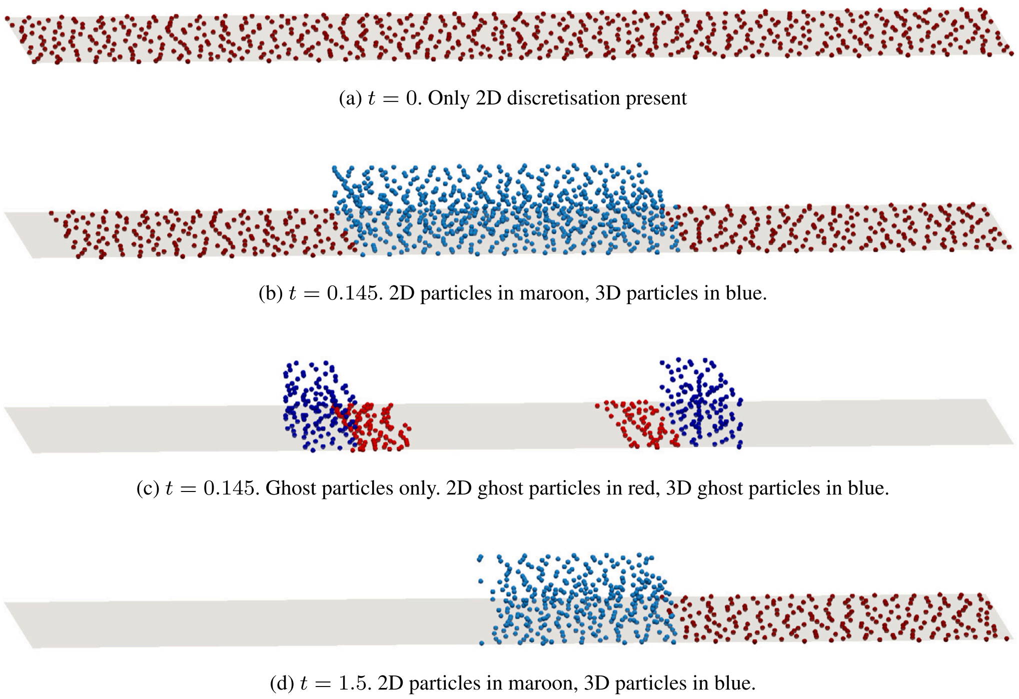
7.1. Advection–diffusion in uniform flow

As the first test case, I consider inviscid flow of a fluid over a flat plate in the absence of gravity. A thin layer of fluid of uniform thickness is considered, with slip velocity boundary conditions on the plate. This test case checks the model transition mechanism (as outlined in Section 5) and the data communication when both models are present beside each other (as outlined in Section 6).

Consider a long plate, with fluid flowing over the surface $[0, 3] \times [0, 1] \times [0]$ in the xy plane. The fluid is discretized alternatingly with the 2D and 3D models, as shown in Fig. 5. An initial velocity of $\vec{v} = (1, 0, 0)^T$ is prescribed throughout the domain, which corresponds to uniform flow from left to right in Fig. 5. Since the flow is inviscid, and without gravity, both models produce a

Table 1
Numerical experiments used to highlight the effectiveness of the introduced model adaptive framework, and the motivation behind each case.

Section	Test case	Comments
Appendix C	Steady fluid in a shallow cylinder	Model transition without flow Tests the model transition, and data mapping algorithms
7.1	Advection–diffusion in uniform flow	Tests the model transition, and data mapping algorithms Tests data communication using ghost particles Convergence tests
7.2	Cleaning jet	Full adaptive model
7.3	Automotive water crossing	Full adaptive model with additional user-defined criteria Industrial test case Simulation time comparison


(a) $t = 0$. Only 2D discretisation present

(b) $t = 0.145$. 2D particles in maroon, 3D particles in blue.

(c) $t = 0.145$. Ghost particles only. 2D ghost particles in red, 3D ghost particles in blue.

(d) $t = 1.5$. 2D particles in maroon, 3D particles in blue.

Fig. 5. Advection–diffusion in uniform flow: Domain discretization at different times for $h = 0.2$. The 2D particles are shown in maroon, and the 3D particles are shown in blue. Ghost particles of each discretization type (Figure c) are shown in a brighter shade to distinguish them from regular particles. Ghost particles are not shown in figures a,b and d. The fluid is moving from left to right in a Lagrangian framework, without any inflow. Thus, there is no fluid on the left of the domain as the simulation progresses. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

constant velocity field equal to the initial velocity. An additional advection–diffusion equation in a Lagrangian framework is solved throughout the domain in both solvers.

$$\frac{DT}{Dt} = \kappa \Delta T \,, \tag{22}$$

for diffusivity κ, material derivative $\frac{D}{Dt} = \frac{\partial}{\partial t} + \vec{v} \cdot \nabla$ for fluid velocity $\vec{v}$, and a temperature field T being transported. Eq. (22) is discretized using a Lagrangian movement step followed by an explicit Euler time integration for the diffusion term, and a GFDM type discretization for the Laplacian operator, as explained above. This test case is split into two parts: first with only advection $\kappa = 0$, followed by a convergence study for advection–diffusion with $\kappa \neq 0$. Both parts consider model transition in both directions: from 2D to 3D and 3D to 2D.

Pure advection

Consider a pure advection case with $\kappa = 0$ in Eq. (22), with the domain and initial velocity as mentioned above. Initially, the entire domain follows the 2D thin film model, see Fig. 5(a). From the first time step onwards, a 3D discretization and model is used in $x \in [1, 2]$, see Fig. 5(b). This corresponds to the middle third of the domain in x direction, as shown in Fig. 5. This implies two fixed interfaces between the 2D and 3D models. However, the use of a Lagrangian framework means that particles entering $x \in [1, 2]$ are transitioned to the 3D model, while particles leaving this region are transitioned to the 2D model in every time step. Thus, the ghost particles in the buffer zone (see Fig. 5(c)) are also recreated at every time step.

The initial velocity $\vec{v} = (1, 0, 0)^T$ corresponds to fluid moving from $x = 0$ to $x = 3$, or left to right in Fig. 5. Since there is no inflow of fluid, and a Lagrangian framework is used, no fluid is present on the left of the domain as the simulation progresses, see Figs. 5(b)–5(d). The initial temperature field is set to 0 everywhere except a small circular region (see Fig. 6(a))

$$T(\vec{x}, t = 0) = \begin{cases} 1, & \text{for } \|(x, y) - (x_0, y_0)\| \leq 0.2 \\ 0, & \text{elsewhere} \end{cases}, \tag{23}$$

where (x, y) are the x and y coordinates of $\vec{x}$, and $x_0 = y_0 = 0.5$. Note that the temperature is constant in the depth direction z. Since the models produce a steady velocity equal to the initial velocity, the analytical solution to the temperature transport is given by Eq. (23) with $x_0 = 0.5 + t$ and $y_0 = 0.5$.

Since the advection is performed in a Lagrangian framework, no numerical diffusion is observed in either the 2D or the 3D model. The numerical results of the temperature profile are shown in Fig. 6. The figure illustrates that the transfer of data between the two models is exact to numerical precision. The final temperature at $t = 2$ exactly matches the analytical solution. This forms a further verification of the data transfer between models during model change. A quantified comparison is presented in the next subsection. Note that in a more general scenario, when a discrete field varies along the depth of the domain, information will be lost while averaging from the 3D model to the 2D model. However, since the temperature field is constant along the depth of the domain in the present test case, there is no information loss due to model transition.

Advection–diffusion

In the pure advection test case considered above, since a Lagrangian framework is being used, there is no impact of derivative computation on the transport equation. I now extend this to a case with $\kappa \neq 0$, i.e. with diffusion of the temperature field. This will test the mechanism of the ghost particles as they work with derivative computation.

A similar setup to the pure advection case is considered, but with a larger domain. Fluid is flowing over the surface $[0, 4] \times [0, 3] \times [0]$ in the xy plane. Model transition follows the same setup as the pure advection case. Initially, the 2D model is applied throughout the domain. Starting from the first time step, the 3D model is applied in the region $x \in [1, 2]$. Due to the Lagrangian movement of the fluid, at every time step, 2D particles crossing $x = 1$ are transitioned to the 3D model, and 3D particles crossing $x = 2$ are transitioned to the 2D model. The same initial velocity of $\vec{v} = (1, 0, 0)^T$ is applied, with inviscid fluid without gravity, resulting in the same uniform and constant velocity as earlier. Since the numerical velocity remains $(1, 0, 0)^T$ in both the pure advection case above and the advection–diffusion test case here, it shows that no spurious effects are introduced through the model transition or the data transfer mechanism using ghost particles. A diffusivity of $\kappa = 5 \times 10^{-3}$ is considered, with the initial temperature as specified in Eq. (23).

Numerical results using the model adaptive framework are compared against a very fine 3D solution with $h = 0.04$ and $N = 644860$ particles at initialization. For the model adaptive simulations, the same resolution is for both the 3D and 2D models. For $h = 0.2$, resulting in $N^{3D} = 2234$ particles and $N^{2D} = 2199$ particles, the model adaptive solution is compared to fine 3D reference solution in Fig. 7. Note that the reference solution is constant along the depth (z) direction. Thus, only the boundary particles are shown in the figure to facilitate a visual comparison. The figure shows a good agreement between the reference and model adaptive solution, despite the use of significantly lesser particles in the model adaptive case.

To quantify the difference in the model adaptive solution and the reference solution, I compute relative mean square and relative maximum errors as

$$\epsilon_2 = \frac{\sum_{i=1}^{N} (T_i - T_I^{\text{ref}})^2}{\sum_{i=1}^{N} (T_I^{\text{ref}})^2}, \tag{24}$$

$$\epsilon_\infty = \frac{\max_i (T_i - T_I^{\text{ref}})}{\max_i (T^{\text{ref}})}, \tag{25}$$

where the summation i is over all particles in the model adaptive solution at the end time of $t = 2$, and I is the closest particle in the reference solution to the particle i. Note that due to the unit velocity and Lagrangian framework, at the end time of $t = 2$, the numerical domain consists only of 2D particles in the region $x \in [2, 4]$, as shown in Fig. 7. A quantified error comparison and convergence study as the model adaptive domain is refined is plotted in Fig. 8 and tabulated in Table 2. These results show a convergence of approximately second order, which matches the expected convergence rate due to the second order accurate GFDM discretization used for derivatives. This once again highlights that no spurious effects are introduced through the ghost particle or model transition mechanism.

For the 3D part of the domain, in the coarsest two resolutions considered, $h = 0.4$ and $h = 0.8$, the height is resolved with insufficient particles. In these cases, all 3D particles are either on the wall or the free surface, where boundary conditions would be applied. Thus, due to the lack of sufficient interior particles in these cases, there are insufficient particles to solve the PDEs. To resolve this, I solve the PDE and the boundary condition (BC) simultaneously on all particles in the 3D phase, PDE $+ \zeta$BC $= 0$, for $\zeta = 0.3$. I refer to [11] for more details on this.

Note that the temperature profile is considered to be the same across the height of the domain in the 3D model. Thus, diffusion occurs in the same directions in the 2D and 3D models, only in and parallel to the xy plane. In a more general case, variation within the height profile would occur in the 3D model, but would be averaged out in the 2D model. Such a difference is not considered in this test case.

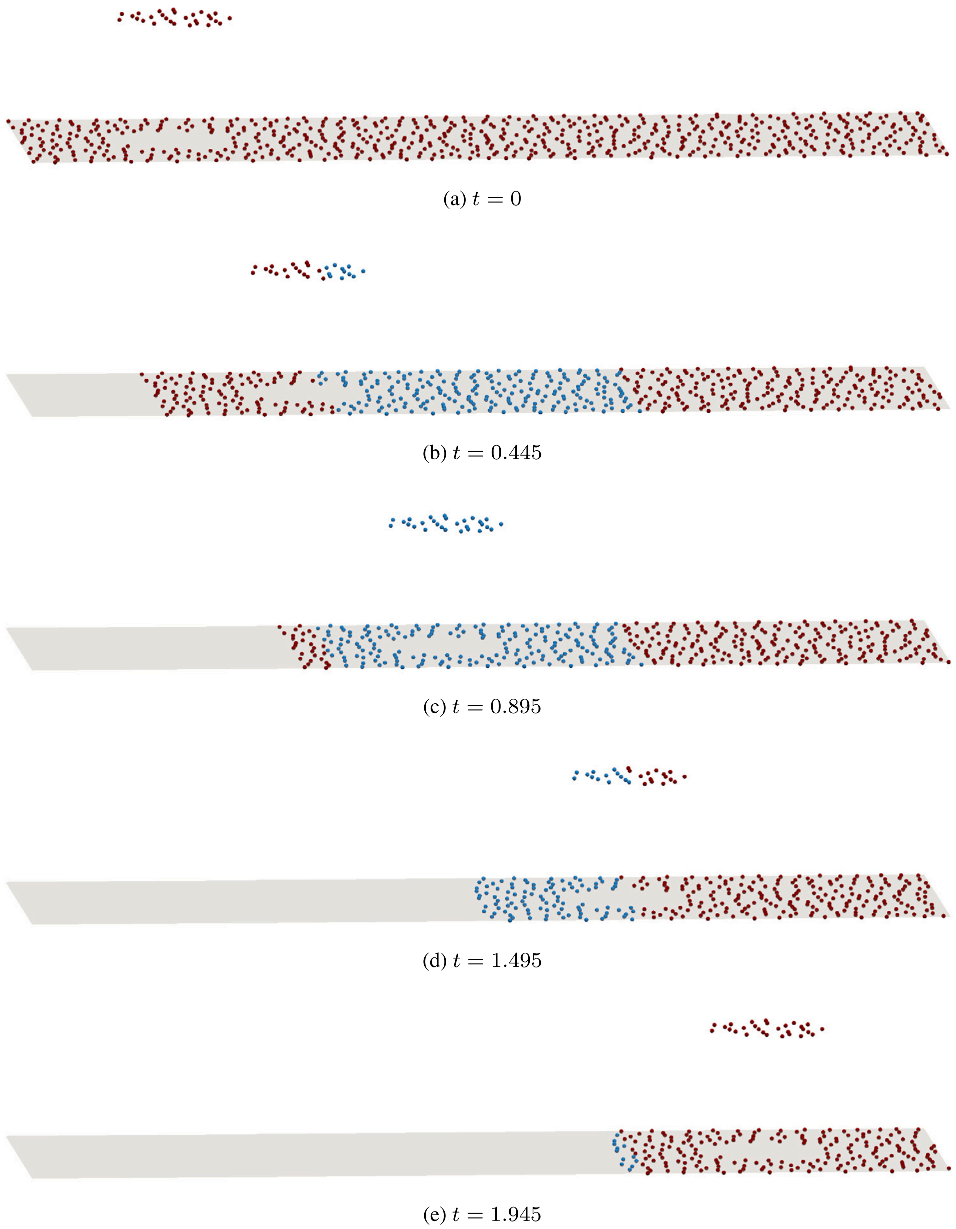


Fig. 6. Advection–diffusion in uniform flow: Temperature profile for the pure advection case. 2D discretization in maroon, 3D discretization in blue. All particles with $T = 0$ are shown on the surface, and all particles with $T > 0$ are raised proportional to the value of T. For the 3D discretization (blue), for the ease of visualizing the solution, only the boundary particles are shown with the temperature indicating the average temperature across the height of the discretization at that location. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

7.2. Cleaning jet

The previous two test cases considered model change in steady flow. I now test the full model adaptive scheme during unsteady flow. Consider a concept cleaning simulation. This test case is motivated by cleaning applications in the food industry, where large trays are cleaned in what is known as “cleaning-in-place” processes. For the concept simulation considered here, I consider a flat surface being “cleaned”, without any contamination on it; and a jet of incoming fluid, as shown in Fig. 9(a). As the jet of fluid hits the surface, it forms a thin layer of fluid. Modelling the thin layer of fluid is essential for cleaning applications, since it reduces the cleaning efficiency due to reduced surface stresses as compared to the hypothetical situation where no fluid film is present. Since the thickness of the film can be much smaller than the characteristic dimension of the jet inlet, the resolution needed for a purely 3D simulation is very fine. Since a thin film approximation is not valid at the inflow, both 2D and 3D models are required for this

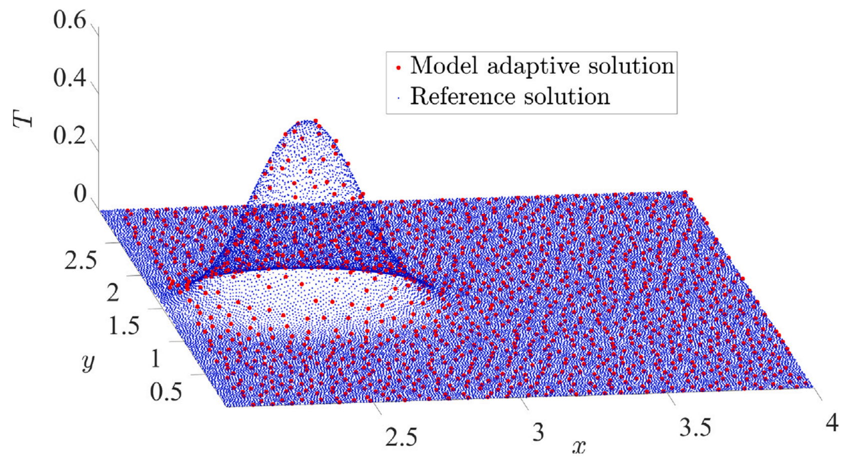


Fig. 7. Advection–diffusion in uniform flow with $\kappa \neq 0$: Comparison of temperature profile at $t = 2$ of the model adaptive method (red) against a fine full 3D reference solution (blue). The case of $h = 0.2$ is shown for the model adaptive solution. See Fig. 8 and Table 2 for a quantified comparison with the reference solution. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

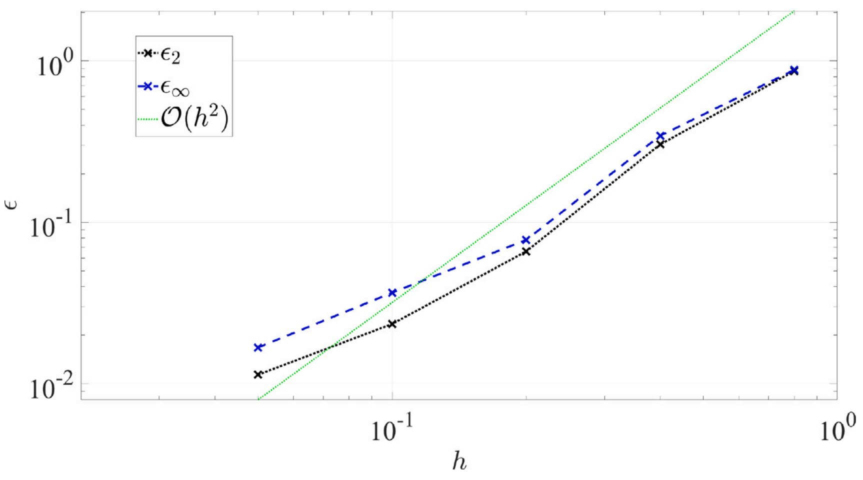


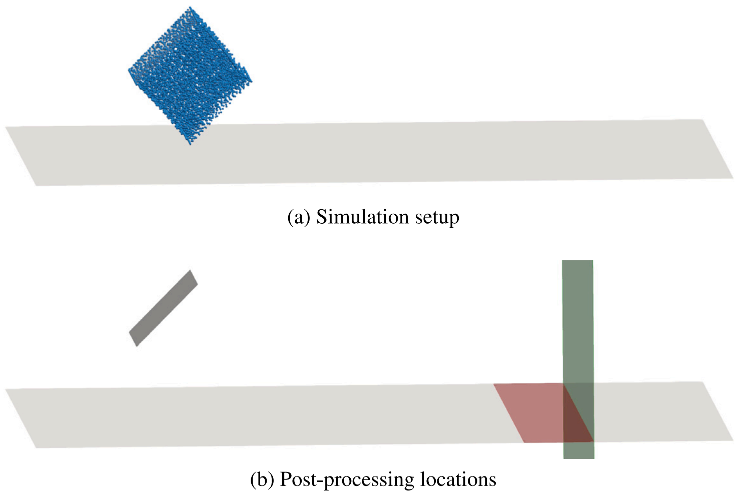
Fig. 8. Advection–diffusion in uniform flow: Convergence of relative errors as the adaptive discretization is refined for the $\kappa \neq 0$ case. Relative L^2 error ϵ_2 (black) and relative L^∞ error ϵ_∞ (blue) compared to a second order convergence rate (green). The errors are tabulated in Table 2. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

Table 2
Advection–diffusion test case: Convergence of error in model adaptive solution when compared to a fine 3D reference solution. h is the resolution used, N^{3D} is the number of 3D particles, and N^{2D} is the number of 2D particles, $N^{\text{total}} = N^{3D} + N^{2D}$ is the total number of particles. All particle numbers correspond to that at the end of the first time step, just after the 3D phase has been created. ϵ_2 and ϵ_∞ (see Eqs. (24),(25)) are the L^2 and L^∞ errors respectively at $t = 2$.

h	N^{3D}	N^{2D}	N^{total}	ϵ_2	ϵ_∞
$h = 0.8$	70	134	204	8.65×10^{-1}	8.79×10^{-1}
$h = 0.4$	330	502	832	3.04×10^{-1}	3.45×10^{-1}
$h = 0.2$	2 234	2 199	4 433	6.61×10^{-2}	7.79×10^{-2}
$h = 0.1$	14 950	8 837	23 787	2.35×10^{-2}	3.67×10^{-2}
$h = 0.05$	126 533	37 867	164 400	1.14×10^{-2}	1.68×10^{-2}

simulation. Furthermore, the regions where the 2D model will be needed cannot be specified a-priori. Thus, this test case forms the perfect application for the model adaptive framework developed in the present work.

First, I have a look at the result qualitatively, to determine if the identification of thin films is done correctly. Fig. 10 shows a comparison of a 3D only simulation with the model adaptive simulation at various time steps. For the model adaptive simulation, the same resolution is used for both the 3D and 2D phases. In the 3D only simulation, the resolution is refined near the surface to capture the flow in the thin fluid regions. For the model adaptive framework, the figure suggests a good behaviour of the PCA-based


(a) Simulation setup

(b) Post-processing locations

Fig. 9. Cleaning jet: Illustration of test case and post-processing locations. Figure (a) shows the inflow (at the top left in each figure) of fluid (blue particles) which will hit the bottom surface being cleaned. Quantification of the results are done using virtual post-processing planes shown in Figure (b). The two quantities of interest studied are the flux of fluid passing through the vertical green plane, and the average normal stresses in the horizontal red region. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

thin film detection method. At a slight distance from where the jet hits the surface, the presence of thin fluid films are detected, and the 3D model gets converted to the 2D model.

To quantify the difference between the 3D only simulation and the hybrid model adaptive results, I measure the fluid flux across a fictitious vertical plane shown in green in Fig. 9(b). The instantaneous flux q through the virtual plane at a time t, and the time cumulative flux Q indicating the total fluid flowing through the plane until time t are measured as

$$q = \int_{\partial\Omega_{\text{virtual}}} \vec{v}\cdot\vec{n}\, dA\,, \tag{26}$$

$$Q = \int_0^t q(\tau)\, d\tau\,. \tag{27}$$

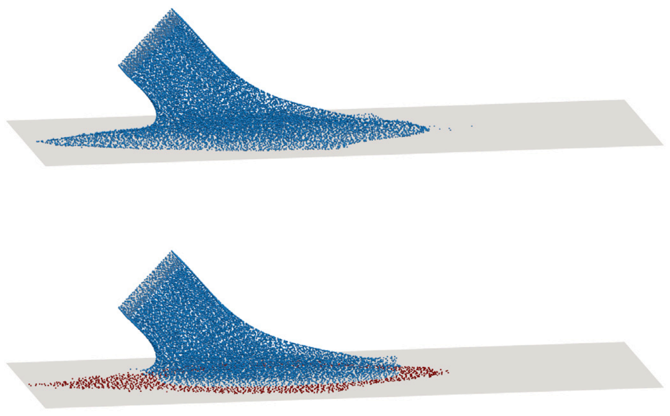
Numerically, q is computed as a sum over all particles crossing the virtual plane. A comparison of results for the 3D and model adaptive simulations, for both the instantaneous and cumulative flux, are shown in Fig. 11. The figure suggests a good match between the model adaptive results and the fine 3D results. For the instantaneous flux, both simulations exhibit a fluctuation around the same mean value of approximately 0.2. The model adaptive results show lesser fluctuations for instantaneous flux than the pure 3D model. This is expected due to the depth averaged flow profile in the 2D model. As the error accumulates, the maximum relative error in Q at the end of the simulation time is only 4.2%.

Since this test case is motivated by a cleaning application, for a further quantified comparison between the 3D and model adaptive simulations, I propose an indicator of how well the fluid will clean the surface. I compare the scalar component of the viscous stress perpendicular to the surface, defined as $\vec{n}^T\mathbf{S}\vec{n}$ for viscous stress $\mathbf{S} = \eta\left(\nabla\vec{v} + (\nabla\vec{v})^T\right)$. This represents a notion of comparison of the cleaning effect of the jet on the surface. Note that this also forms the normal component of the traction. Rather than considering the normal traction at a single location, I consider an average over a small region of the surface, as shown in red in Fig. 9(b). The average value of $\vec{n}^T\mathbf{S}\vec{n}$ in this region is shown in Fig. 12 for both 3D and model adaptive simulations. Once again, both results are very similar. The initial value of 0 indicates that fluid has not yet reached this post-processing zone. The highest deviation between the two approaches is observed when the fluid first reaches this zone, where the maximum deviation is 8.1% and there are high fluctuations in the normal traction.

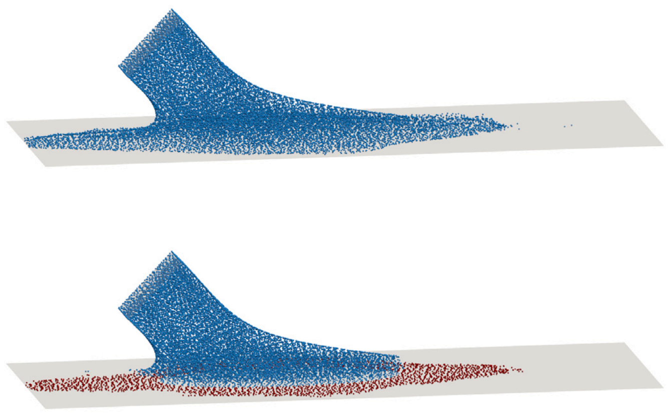
Since the focus of the present work is on the model adaptivity, several important fluid effects for these applications have not been considered. Most importantly, surface tension and wetting angle effects have been neglected.

7.3. Application: Automotive water crossing

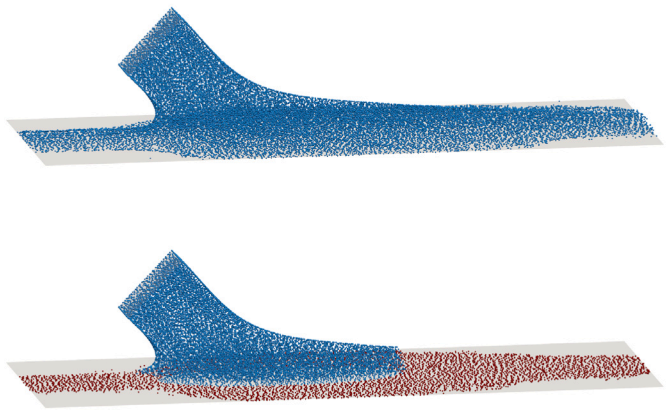
I now present an application of the model adaptive framework introduced in the present work on an industrial test case. Consider a car crossing a shallow pool of water of constant depth, see [19,48]. The study of water wading depth, or the depth of water through which a car can safely cross, has become very important in vehicular design. As shown in Fig. 13, the simulation domain consists of a long shallow pool of water, about 3 times the length of the car. The crucial part of the simulation domain is the fluid just around the car. Fluid in the far field plays a negligible role on the main quantities of interest: the traction on the wheels, and the water spray patterns. For more details on this application, and for mass conservation considerations, see [19].



(a) $t = 0.21$s. 3D only (top), and model adaptive (bottom). 3D model and dsicretization shown in blue, 2D in maroon.

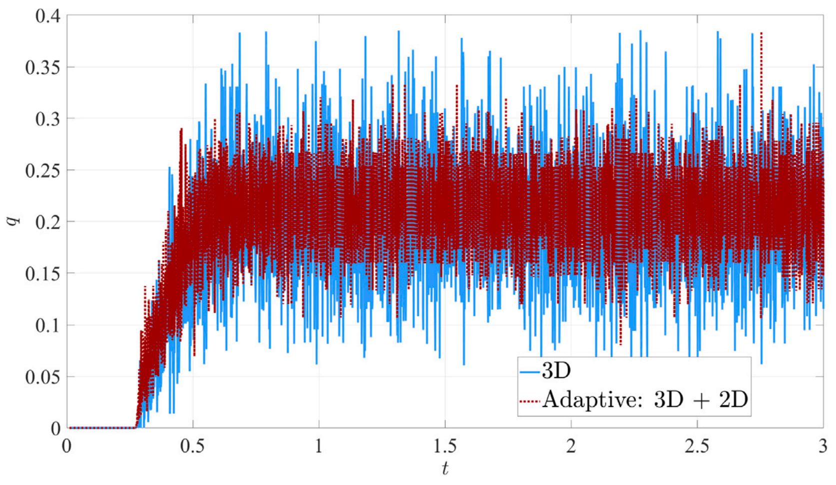


(b) $t = 0.26$s. 3D only (top), and model adaptive (bottom). 3D model and dsicretization shown in blue, 2D in maroon.

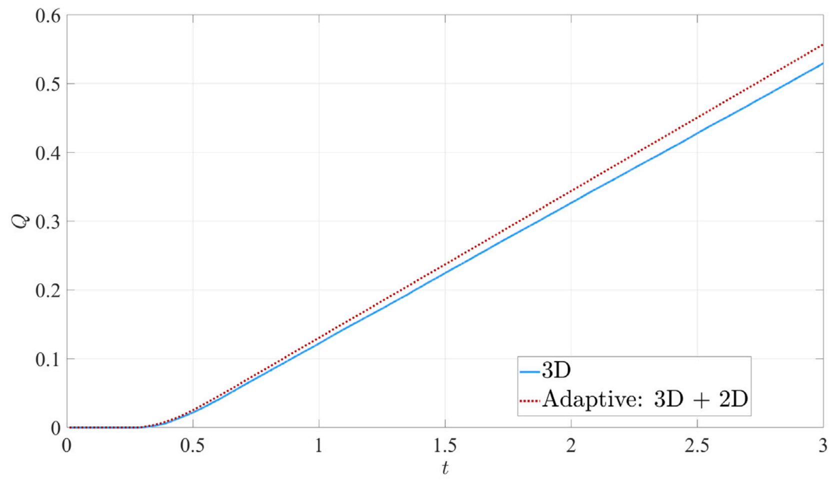


(c) $t = 0.4$s. 3D only (top), and model adaptive (bottom).3D model and dsicretization shown in blue, 2D in maroon.

Fig. 10. Cleaning jet: Visual results comparisons between the 3D only simulations and the model adaptive (3D+2D) simulations. Particles in blue indicate the 3D phase, while particles in maroon indicate the 2D phase. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)



(a) Instantaneous flux q, see Eq. (26).



(b) Cumulative flux Q, see Eq. (27).

Fig. 11. Cleaning jet: Comparison of fluid flux through a virtual plane for the fine 3D and model adaptive (3D+2D) simulations. See Fig. 9(b) for the location of the virtual post-processing plane.

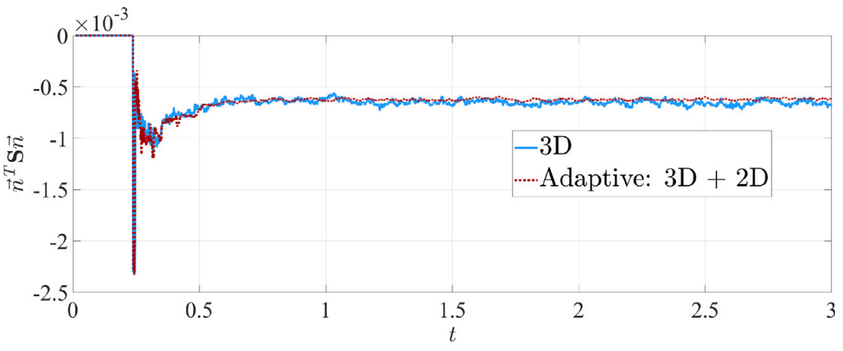


Fig. 12. Cleaning jet: Comparison of cleaning effect of the 3D and model adaptive (3D+2D) simulations, given by the scalar component of the viscous stress perpendicular to the surface.

Table 3
Automotive water crossing: Number of points N, and computational time required t_{comp} (in minutes) for a simulated time of 1 second. All simulation are run in parallel using 4 MPI processes. The number of points is reported at $t = 0.5$.

h	3D only		Model adaptive (3D+2D)			
	N^{3D}	t_{comp}	N^{3D}	N^{2D}	N^{total}	t_{comp}
$h = 0.2$	30 497	12	15 956	6 453	22 409	7
$h = 0.15$	63 468	33	30 488	11 086	41 574	16
$h = 0.1$	195 201	181	81 202	23 351	104 553	56

For such a scenario, adaptive refinement significantly helps in speeding up simulations. However, a significant drawback in meshfree methods is the lack of anisotropic resolutions. This introduces a minimum resolution needed to resolve the thickness of the film, which is needed throughout the domain. To overcome this drawback in using adaptive refinement, I propose the use of adaptive model selection. The automatic detection of model adaptivity presented in Section 4 is supplemented with a user-defined criterion based on the distance of the fluid to the car. Particles sufficiently close to the car are solved with the 3D model, while those far away are solved with the 2D model. As a result, when the car moves across the channel (from left to right in Fig. 13), at every time step, 2D particles coming near the car are transitioned to the 3D model, and 3D particles far away from the car are shifted to the 2D model.

The height of the pool is 12 cm, and the car crosses the pool with a uniform speed of 40 km/h. The car is assumed to be a rigid body, except the wheels and tyres, each of which rotate about their centre point with an angular velocity that matches the prescribed liner velocity of the car. Snapshots of the numerical results of the 3D simulation and model adaptive simulation are shown in Fig. 13. The figure shows that both methods produce very similar results. The fluid is at rest in front of the car. Thus, no information is lost by applying the 2D model ahead of the car (Fig. 13 shows where the 2D model is being used). As a result, the 3D region in the model adaptive simulation perfectly matches the corresponding region in the 3D only simulation. All quantities of interest measured around the car are exactly the same in both simulations. This was verified by checking the evolution of the pressure of the water on the wheels, and the spray of water hitting the car, for three resolutions each.

The results of the two approaches (3D and model adaptive) only differ in the wake of the car. It is important to note here that typical quantities of interest in water crossing applications, like the traction on the wheels and the fluid spray on the car, are measured only around the car, with no influence of the flow patterns in the wake.

The number of particles in the discrete domain and the simulation time for simulating 1 second of physical time is tabulated in Table 3 for three different resolutions each of the 3D only approach and the model adaptive approach. In the model adaptive simulation, the same resolution is used for the both the 3D and 2D phases. Table 3 shows that the coarsest simulation considered results in a speed-up of a factor of 1.7 for the model adaptive approach. As the number of particles increases, the gap in the simulation time between the approaches increases. For the finest simulation of $h = 0.1$, the model adaptive framework introduced in this work results in a speed-up of a factor of 3.2, while maintaining the same flow results around the car.

8. Limitations and open questions

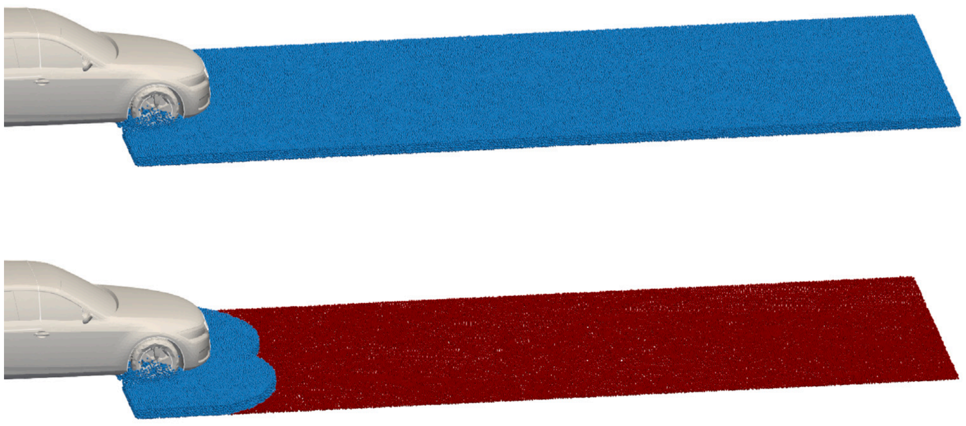
While the results shown in Section 7 suggest that the model adaptive framework presented here is very promising, there are several limitations and open questions that must be discussed.

Model differences

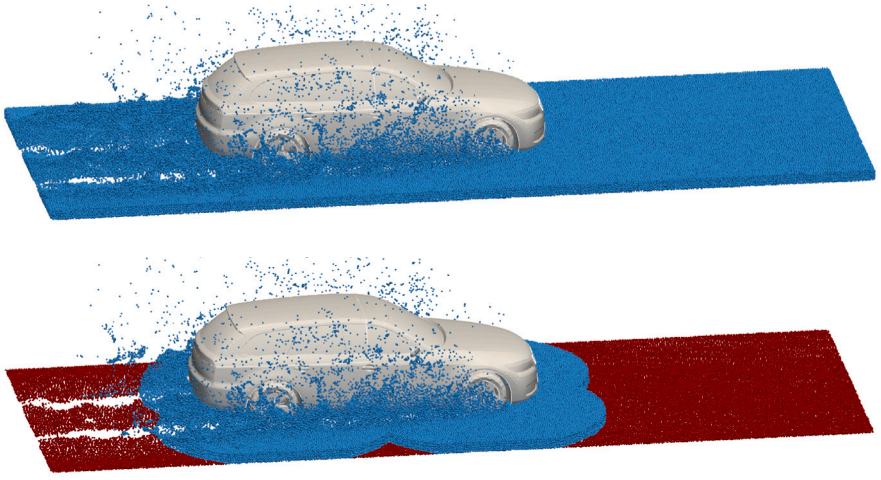
- The differences in the assumptions of the two models are crucial in understanding the capabilities of the presented model adaptive framework. For instance, the 2D thin film model is derived under the assumption of a quadratic velocity profile [12]. However, while making the 2D to 3D transition, the presented framework enforces a uniform velocity profile (with slip boundary conditions), rather than a quadratic profile.
- In a general scenario, information is lost when depth averaging from the 3D model to the 2D model. This raises the question as to when does this transition result in an error accumulation that is unacceptable. Further tests need to be done to properly test the limits of this work.

Detecting model transition

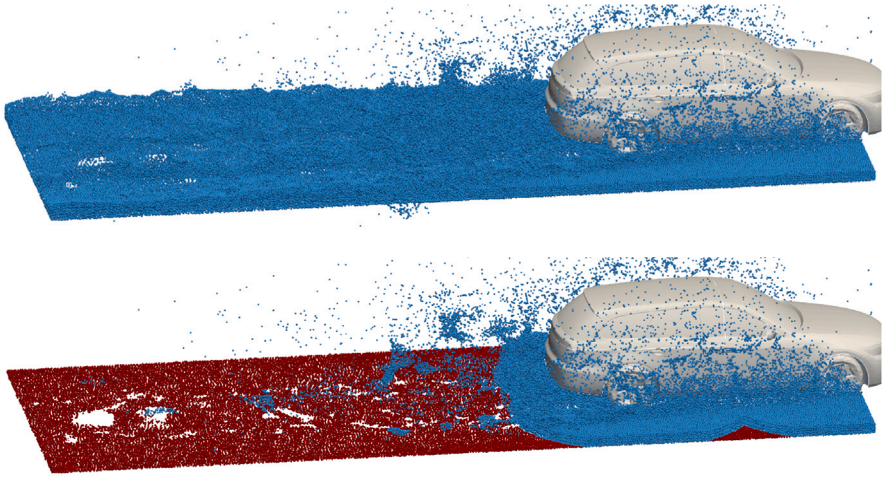
- For the identification of thin layers, a further study is needed to determine the best approach. The local PCA presented in Section 4 forms a sufficient first model, but requires further testing to determine its limits and optimal parameter values for $\epsilon_\lambda^{\max}$ and θ^*. We also need quantifiable tests to compare different methods of thin film detection. Further testing of the PCA approach is needed before a claim can be made concerning the robustness of the automatic model transition detection. Weighted PCA approaches [49], and related work on the rupture of thin films [50] should be investigated as alternatives to the PCA approach presented here.
- In the context of h-adaptivity, the use of error indicators and estimators to decide when to refine or coarsen a discretization has been shown to be a very robust and effective approach (see, for example, [51,52]). This raises the question if an error-based approach to change models during model adaptivity could be devised to either replace or compliment the heuristic based approach proposed here.



(a) $t = 0.07$s. 3D only (top), and model adaptive (bottom). 3D model and discretization shown in blue, 2D in maroon.



(b) $t = 0.57$s. 3D only (top), and model adaptive (bottom). 3D model and discretization shown in blue, 2D in maroon.



(c) $t = 1$s. 3D only (top), and model adaptive (bottom). 3D model and discretization shown in blue, 2D in maroon.

Fig. 13. Automotive water crossing: Illustration of user-defined transition criteria for transitioning between the 2D thin film and 3D bulk models. Here, I use a minimum distance from the four car wheels as the criterion for model change. particles in blue indicate the 3D phase, while particles in maroon indicate the 2D phase. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

Data communication

While the buffer zone approach using ghost particles introduced in Section 6 works very well, as shown in Section 7, a significant issue of this is that is hampers code maintainability. The addition of ghost particles handling to every instance of derivative computation makes the code cumbersome to maintain, especially when there are multiple developers. I thus wish to investigate alternative approaches to data communication without the use of ghost particles.

9. Conclusion

I presented a novel adaptive procedure to couple a 3D Navier–Stokes model with a pseudo-2D thin film flow model. By analysing the principal components of a local neighbourhood, the proposed algorithm detected when and where a thin film model would be applicable. Lagrangian particles were used as the domain discretization in both models. When the need for model change from 3D to 2D or vice versa is detected for a particle, all required data is transferred between models, and the required change in discretization (3D vs 2D) is also performed automatically. When particles governed by different models lie beside each other, data is communicated using a two-way coupling between them using a buffer region where both models are solved. The buffer region is populated with ghost particles that assist in data communication. Due to the dynamic nature of the typical applications of the model adaptive framework, ghost particles need to be created at every time step. Numerical results show a good comparison of the adaptive framework with a pure 3D bulk flow situation, with a significant time speed up. My future work in this direction will be towards investigating the limitations listed in Section 8, and extending the work to flow on curved and moving surfaces.

While this work only considered the adaptive transition from a 3D flow model to a 2D one. This lays the foundation for an adaptive selection between any two models of the same phenomenon, in a vast variety of scenarios. This forms a crucial part of my ongoing work.

CRediT authorship contribution statement

Pratik Suchde: Writing – review & editing, Writing – original draft, Visualization, Validation, Software, Project administration, Methodology, Investigation, Funding acquisition, Formal analysis, Conceptualization.

Declaration of competing interest

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

Data availability

Data will be made available on request.

Acknowledgements

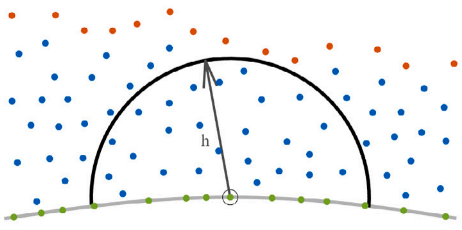
Pratik Suchde would like to acknowledge partial support from the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie Actions grant agreement No. 892761 "SURFING". Pratik Suchde would also like to acknowledge funding from the Institute of Advanced Studies, University of Luxembourg, under the AUDACITY programme grant "ADONIS".

Appendix A. Detecting 3D to 2D transition: Resolution criterion

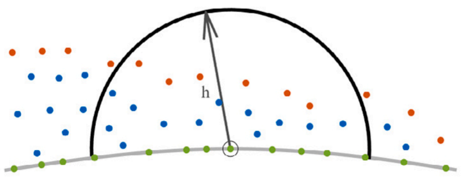
In this appendix, I elaborate on the resolution criterion (introduced in Section 4.1.1) for detecting when a 3D particle should be transitioned to the 2D model. The notion of checking the resolution is evaluated by looking for free surface particles close to a wall.

- If the 3D particle i has interior neighbours, but no free surface neighbours, then no model transition will be done. All neighbours of such a particle would either be interior particles or wall boundary particles, see Fig. 14(a), and particles are present throughout the neighbourhood. Thus, the effective height of a theoretical thin film model at this location would be larger than the interaction radius: $h_i < H_{\text{film}}(\vec{x}_i)$, making it impossible to prescribe a local condition for model transition.
- If particle i has at least one free surface neighbour, then it is flagged for further checks to see if a thin film model is applicable, see Fig. 14(b). This implies that the bulk model resolution may not be sufficiently fine to capture the fluid behaviour at that location: $h_i > H_{\text{film}}(\vec{x}_i)$.
- If particle i only has wall boundary neighbours, with no interior or free surface neighbours, it is flagged for model transition to the thin film model, see Fig. 14(c). We have $h_i \gg H_{\text{film}}(\vec{x}_i)$.

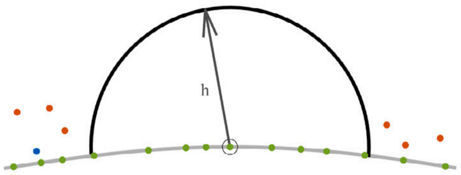
As explained in Section 2.3, the minimum and maximum inter-particle distance are a fixed ratio of the interaction radius. Thus, h is also an indicator of the resolution of the point cloud. As a result, the criterion of looking for free surface and interior particle neighbours within the interaction radius is inherently based on the resolution.



(a) The particle being evaluated has only interior and wall boundary neighbours, with no free surface neighbours. No transition to thin film will occur.



(b) The particle being evaluated has free surface neighbours (in addition to interior and wall boundary neighbours). The PCA criterion will be evaluated to determine if transition to thin film will occur.



(c) The particle being evaluated has no interior or free surface neighbours. This particle will be changed to the thin film model without the need to evaluate the PCA criterion.

Fig. 14. Resolution based criterion for detecting bulk to thin film transition. Wall particles are marked in green, interior particles in blue, and free surface particles in orange. The wall particle at which the bulk to thin film transition criterion is being evaluated is marked with an additional black circle, with its neighbourhood highlighted. The wall is marked in grey. Note that throughout this work a 3D bulk model and 2D thin film model is considered. This figure shows a lower dimensional schematic for the ease of visualization. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

Remark. In contrast to the meshfree collocation approach used here, in several meshfree methods like Smoothed Particle Hydrodynamics (SPH), there is usually no fixed relation between the interaction radius and the minimum particle spacing. Thus, this resolution criteria would need to be modified if this framework is to be applied to an SPH based bulk discretization.

Appendix B. Creating ghost particles

The creation of ghost particles (see Section 6.1) follows a three-step procedure.

1. In the first step, particles that will create ghost particles are flagged, see Fig. 4(b).
2. Subsequently, for all flagged particles, a ghost particle is created at the same location, see Fig. 4(c).
3. In the final step, model transition is performed on these ghost particles, see Fig. 4(d).

For the first step, consider a particle i and all nearby particles within a distance of h from i. If there exists a nearby particle j which is governed by a different model than i, then both i and j are flagged to create ghost particles. Thus, if i is a 3D particle, all neighbouring 2D particles are flagged, and vice versa. In the second step, for each flagged particle j, a ghost particle is created at the same location as j with the same model as j. This forms a set of intermediate ghost particles. Finally, model transition is performed on all intermediate ghost particles, as done in Section 5, to get the final set of ghost particles. This procedure is illustrated in Fig. 4.

To further describe this procedure, consider a 3D particle i. The creation of ghost particles near i starts by flagging all 2D particles within a distance of h from i. Each of these flagged 2D particles create co-incident 2D ghost particles. Finally, all 2D ghost particles are transitioned to 3D ghost particles, by adding further 3D ghost particles in the normal direction and appropriate data mapping, as per Section 5.2. These newly created 3D ghost particles contain information from the 2D model in the buffer region, and will be used in all approximations performed at particle i. Similarly, for a 2D particle i, all 3D neighbours are flagged as sources for ghost particles. The flagged 3D particles create 3D ghost particles, which are finally transitioned to 2D with the discretization adaptivity and data mapping procedure laid down in Section 5.1. This includes the deletion of all interior and free surface ghost particles, with their data mapped to the corresponding wall boundary ghost particles before its model is changed to the 2D thin film model.

Appendix C. Results: Steady fluid in a shallow cylinder

As an additional validation test case, I run the model transition algorithm with the PDE solvers switched off. Consider a cylinder of unit radius, filled partially with a stationary fluid. The resolution of the domain is varied such that it triggers the model transition criteria in the entire domain from 3D to 2D, and then vice versa. Since the entire domain is solved either with a 3D model, or a 2D

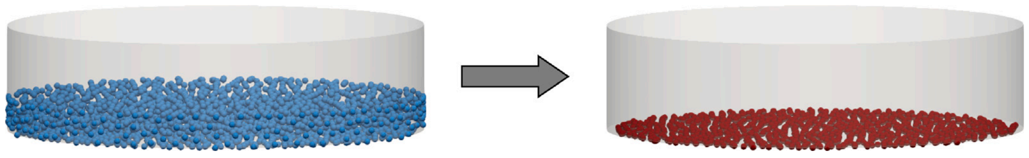


Fig. 15. Steady fluid in a shallow cylinder: Transition of a 3D model (left) to a 2D model (right).

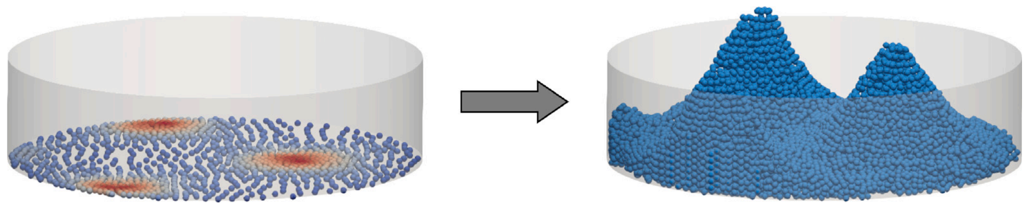


Fig. 16. Steady fluid in a shallow cylinder: Transition of a 2D model (left) to a 3D model (right). In the 2D figure (left), the colour indicates the computed height function. A comparison of the computed height function in the 2D model and the free surface in the 3D model is shown in Fig. 17. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

model, ghost particles and buffer zones do not play a role here. This case only tests the model transition detection, the discretization change during the model transition, and the data mapping between the models during model transition. The fluid considered is water, with density $\rho = 1000$ kg/m^3 and viscosity $\eta = 10^{-3}$ Pa s.

3D to 2D

Consider the cylinder filled to a height of 0.15 m, and initialized with a 3D discretization, see Fig. 15. A resolution of $h = 0.2$ is used for the initial discretization, which results in $N = 4758$ particles in the domain. As the simulation starts, the resolution of the bulk model is changed to $h = 0.3$. This results in the resolution becoming too coarse to capture the layer of fluid. I observe that the PCA criterion from Section 4.1.2 gets triggered at all wall boundary particles, resulting in a model transition to the 2D model. Thus, all particles transition to 2D at the same time. Fig. 15 shows the discretization after the model transition is complete. The figure shows that all interior and free surface particles are deleted, as desired. Furthermore, the boundary particles on the vertical walls of the cylinder do not transition to a thin film model, since they fail the eigenvector criterion of Section 4.1.3. These wall particles are subsequently deleted during the transition process of the wall particles on the base of the cylinder.

Fig. 15 shows the discretization of the domain before and after transition. To verify the data transfer between the models, I check the following. The velocity of the fluid is completely at rest before and after model transition. The total mass and volume are the same up to numerical precision for both models. The analytically expected volume to be occupied by the domain is $0.15\pi \approx 0.4712$. Numerically, the volume occupied by the 3D discretization is $\sum_i V_i = 0.4631$. The slight discrepancy from the analytical volume (under 2%) is expected due to the particle based discretization, see [19] for more details. As expected from the data mapping algorithm, the total numerical volume occupied by the 2D model after transition matches this to numerical precision. Similarly, the total mass in the two models match up to numerical precision, $\sum_i m_i = 0.4631 \times 10^3$. This verifies that the mapping from the interior and free surface particles being deleted to the wall particles works as intended, and that the conservative data transfer during 3D to 2D discretization works correctly.

2D to 3D

Now consider the same problem with the model transition happening in the other direction. The domain is initialized with a 2D discretization. Unlike the previous case, to increase the complexity of the problem, I initialize the film with a spatially varying thickness, as shown in Fig. 16. The resolution of the 3D model is reduced throughout the domain to trigger the model transition criterion (Section 4.2) throughout the domain. Once again, I observe that the entire domain transitions to the 3D model at the same time.

Fig. 16 shows the discretization of the domain before and after transition, while Fig. 17 shows the comparison of the free surface profiles in each model. This considers the top most free surface profile in the 3D model (maroon) and the computed height function in the 2D model (blue) overlaid on each other. The figure shows that both models produce the same free surface profile. Since the PDE solvers are switched off in this test case, the spatially varying height of the fluid does not result in a tangential velocity. The fluid maintains the initially prescribed 0 velocity, both before and after model transition. Once again, the total mass and volume are the same up to numerical precision for both models, with the total numerical volume $\sum_i V_i = 0.5592$, and the total numerical mass $\sum_i m_i = 0.5592 \times 10^3$. This highlights the conservative nature of the data mapping during model transition.

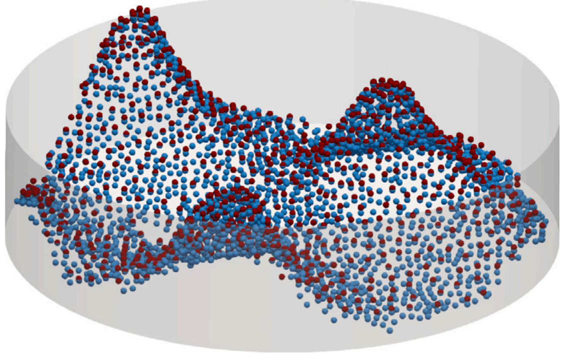


Fig. 17. Steady fluid in a shallow cylinder: Transition of the 2D model to the 3D model. Comparison of the computed height profile in the 2D model (maroon) and the free surface of the 3D model (blue). The individual domain discretization of each model are shown in Fig. 16. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

References

[1] C.B. Vreugdenhil, Numerical Methods for Shallow-Water Flow, vol. 13, Springer Science & Business Media, 2013.

[2] A.L. Bertozzi, M. Pugh, The lubrication approximation for thin viscous films: Regularity and long-time behavior of weak solutions, Commun. Pure Appl. Math. 49 (2) (1996) 85–123.

[3] S. Chakraborty, P.-K. Nguyen, C. Ruyer-Quil, V. Bontozoglou, Extreme solitary waves on falling liquid films, J. Fluid Mech. 745 (2014) 564–591.

[4] T.-P. Fries, Higher-order surface FEM for incompressible Navier-Stokes flows on manifolds, Int. J. Numer. Methods Fluids 88 (2) (2018) 55–78.

[5] C.P. Martin-Linares, Y.M. Psarellis, G. Karapetsas, E.D. Koronaki, I.G. Kevrekidis, Physics-agnostic and physics-infused machine learning for thin films flows: modelling, and predictions from small data, J. Fluid Mech. 975 (2023) A41.

[6] W. Rohlfs, M. Rietz, B. Scheid, WaveMaker: The three-dimensional wave simulation tool for falling liquid films, SoftwareX 7 (2018) 211–216.

[7] G. Ling, J. Matsumoto, K. Kashiyama, A two-way coupling 2D-3D hybrid finite element numerical model using overlapping method for tsunami simulation, Internat. J. Numer. Methods Fluids 95 (11) (2023) 1732–1755.

[8] F. Mintgen, M. Manhart, A bi-directional coupling of 2D shallow water and 3D Reynolds-averaged Navier–Stokes models, J. Hydraul. Res. 56 (6) (2018) 771–785.

[9] S. Pan, R. Nomura, G. Ling, S. Takase, S. Moriguchi, K. Terada, Variable passing method for combining 3D MPM–FEM hybrid and 2D shallow water simulations of landslide-induced tsunamis, Internat. J. Numer. Methods Fluids 96 (1) (2024) 17–43.

[10] S. Pan, Y. Yamaguchi, A. Suppasri, S. Moriguchi, K. Terada, MPM–FEM hybrid method for granular mass–water interaction problems, Comput. Mech. 68 (1) (2021) 155–173.

[11] P. Suchde, J. Kuhnert, S. Tiwari, On meshfree GFDM solvers for the incompressible Navier–Stokes equations, Comput. & Fluids 165 (2018) 1–12.

[12] A.S. Bharadwaj, J. Kuhnert, S.P. Bordas, P. Suchde, A discrete droplet method for modelling thin film flows, Appl. Math. Model. 112 (2022) 486–504.

[13] J. Behrens, Towards model-adaptivity: Localized non-hydrostatic wave modeling, in: Geophysical Research Abstracts, Vol. 21, 2019.

[14] M. Braack, A. Ern, A posteriori control of modeling errors and discretization errors, Multiscale Model. Simul. 1 (2) (2003) 221–238.

[15] P. Suchde, J. Kuhnert, S. Schröder, A. Klar, A flux conserving meshfree method for conservation laws, Internat. J. Numer. Methods Engrg. 112 (3) (2017) 238–256.

[16] P. Suchde, J. Kuhnert, A fully Lagrangian meshfree framework for PDEs on evolving surfaces, J. Comput. Phys. 395 (2019) 38–59.

[17] R. Löhner, E. Onate, An advancing front point generation technique, Commun. Numer. Methods Eng. 14 (12) (1998) 1097–1108.

[18] J. Slak, G. Kosec, On generation of node distributions for meshless PDE discretizations, SIAM J. Sci. Comput. 41 (5) (2019) A3202–A3229.

[19] P. Suchde, C. Leithäuser, J. Kuhnert, S. Bordas, Volume and mass conservation in Lagrangian meshfree methods, 2023, arXiv preprint arXiv:2303.13410.

[20] A.J. Chorin, Numerical solution of the Navier-Stokes equations, Math. Comput. 22 (104) (1968) 745–762.

[21] P. Suchde, J. Kuhnert, Point cloud movement for fully Lagrangian meshfree methods, J. Comput. Appl. Math. 340 (2018) 89–100.

[22] P. Suchde, T. Jacquemin, O. Davydov, Point cloud generation for meshfree methods: An overview, Arch. Comput. Methods Eng. 30 (2) (2023) 889–915.

[23] I. Michel, T. Seifarth, J. Kuhnert, P. Suchde, A meshfree generalized finite difference method for solution mining processes, Comput. Part. Mech. 8 (3) (2021) 561–574.

[24] C. Drumm, S. Tiwari, J. Kuhnert, H.-J. Bart, Finite pointset method for simulation of the liquid - liquid flow field in an extractor, Comput. Chem. Eng. 32 (12) (2008) 2946–2957.

[25] B. Seibold, M-Matrices in Meshless Finite Difference Methods (Ph.D. thesis), University of Kaiserslautern, Germany, 2006.

[26] J. Benito, F. Urena, L. Gavete, Influence of several factors in the generalized finite difference method, Appl. Math. Model. 25 (12) (2001) 1039–1053.

[27] X. Rao, An upwind generalized finite difference method (GFDM) for meshless analysis of heat and mass transfer in porous media, Comput. Part. Mech. 10 (3) (2023) 533–554.

[28] Z. Zheng, X. Li, Theoretical analysis of the generalized finite difference method, Comput. Math. Appl. 120 (2022) 1–14.

[29] T. Jacquemin, S. Tomar, K. Agathos, S. Mohseni-Mofidi, S.P. Bordas, Taylor-series expansion based numerical methods: a primer, performance benchmarking and new approaches for problems with non-smooth solutions, Arch. Comput. Methods Eng. 27 (2020) 1465–1513.

[30] P. Suchde, J. Kuhnert, A meshfree generalized finite difference method for surface PDEs, Comput. Math. Appl. 78 (8) (2019) 2789–2805.

[31] A.S. Bharadwaj, E. Thiel, P. Suchde, A Lagrangian meshfree model for solidification of liquid thin-films, Comput. & Fluids (2024) 106267.

[32] S. Marras, K.T. Mandli, Modeling and simulation of tsunami impact: a short review of recent advances and future challenges, Geosciences 11 (1) (2020) 5.

[33] M. Masó, A. Franci, I. De-Pouplana, A. Cornejo, E. Oñate, A Lagrangian–Eulerian procedure for the coupled solution of the Navier–Stokes and shallow water equations for landslide-generated waves, Adv. Model. Simul. Eng. Sci. 9 (1) (2022) 15.

[34] J. Shlens, A tutorial on principal component analysis, 2014, arXiv preprint arXiv:1404.1100.

[35] H. Hoppe, T. DeRose, T. Duchamp, J. McDonald, W. Stuetzle, Surface reconstruction from unorganized points, SIGGRAPH Comput. Graph. 26 (2) (1992) 71–78.

[36] J. Liang, H. Zhao, Solving partial differential equations on point clouds, SIAM J. Sci. Comput. 35 (3) (2013) A1461–A1486.
[37] O. Awile, F. Büyükkeçeci, S. Reboux, I.F. Sbalzarini, Fast neighbor lists for adaptive-resolution particle simulations, Comput. Phys. Comm. 183 (5) (2012) 1073–1081.
[38] P. Lu, S. Guo, Y. Shu, B. Liu, P. Li, W. Cao, K. Jiang, A local search scheme in the natural element method for the analysis of elastic-plastic problems, Adv. Eng. Softw. 176 (2023) 103403.
[39] J. Onderik, R. Durikovic, Efficient neighbor search for particle-based fluids, J. Appl. Math. Stat. Inform. 4 (1) (2008) 29–43.
[40] M. Mehl, B. Uekermann, H. Bijl, D. Blom, B. Gatzhammer, A. Van Zuijlen, Parallel coupling numerics for partitioned fluid–structure interaction simulations, Comput. Math. Appl. 71 (4) (2016) 869–891.
[41] W.-B. Liu, D.-J. Ma, J.-Z. Qian, M.-Y. Zhang, A.-M. He, N.-S. Liu, P. Wang, High-accuracy three-dimensional surface detection in smoothed particle hydrodynamics for free-surface flows, Comput. Phys. Comm. 290 (2023) 108789.
[42] Y. Lu, A.-k. Hu, Y.-c. Liu, A finite pointset method for the numerical simulation of free surface flow around a ship, J. Mar. Sci. Technol. 21 (2016) 190–202.
[43] F.R. Saucedo-Zendejo, E.O. Reséndiz-Flores, J. Kuhnert, Three-dimensional flow prediction in mould filling processes using a GFDM, Comput. Part. Mech. 6 (3) (2019) 411–425.
[44] S. Tiwari, J. Kuhnert, Particle method for simulation of free surface flows, in: Hyperbolic Problems: Theory, Numerics, Applications: Proceedings of the Ninth International Conference on Hyperbolic Problems Held in CalTech, Pasadena, March 25–29, 2002, Springer, 2003, pp. 889–898.
[45] P.R. Budarapu, X. Zhuang, T. Rabczuk, S.P. Bordas, Multiscale modeling of material failure: Theory and computational methods, Adv. Appl. Mech. 52 (2019) 1–103.
[46] H. Talebi, M. Silani, S.P. Bordas, P. Kerfriden, T. Rabczuk, Molecular dynamics/XFEM coupling by a three-dimensional extended bridging domain with applications to dynamic brittle fracture, Int. J. Multiscale Comput. Eng. 11 (6) (2013).
[47] L. Veltmaat, F. Mehrens, H.-J. Endres, J. Kuhnert, P. Suchde, Mesh-free simulations of injection molding processes, Phys. Fluids 34 (3) (2022).
[48] H. Zhang, X. Li, K. Feng, M. Liu, 3D large-scale SPH modeling of vehicle wading with GPU acceleration, Sci. China Phys. Mech. Astron. 66 (10) (2023) 104711.
[49] E. Castillo, J. Liang, H. Zhao, Point cloud segmentation and denoising via constrained nonlinear least squares normal estimates, in: M. Breuß, A. Bruckstein, P. Maragos (Eds.), Innovations for Shape Analysis: Models and Algorithms, Springer Berlin Heidelberg, Berlin, Heidelberg, 2013, pp. 283–299.
[50] L. Chirco, J. Maarek, S. Popinet, S. Zaleski, Manifold death: a volume of fluid implementation of controlled topological changes in thin sheets by the signature method, J. Comput. Phys. 467 (2022) 111468.
[51] T. Jacquemin, P. Suchde, S.P. Bordas, Smart cloud collocation: geometry-aware adaptivity directly from CAD, Comput. Aided Des. 154 (2023) 103409.
[52] O.C. Zienkiewicz, The background of error estimation and adaptivity in finite element computations, Comput. Methods Appl. Mech. Engrg. 195 (4–6) (2006) 207–213.