Eleventh International Conference on Computational Fluid Dynamics (ICCFD11), Maui, HI, USA, July 11-15, 2022

ICCFD11-2022-1301

Developing a Modern CFD Framework with Parallel Algorithms and Mesh Adaption

J. McKee, Y. Mileyko, A. Fisher, and A. Koniges
Corresponding author: koniges@hawaii.edu

University of Hawaiʻi at Manōa, HI, USA

Abstract: We discuss recent advances in a Computational Fluid Dynamics (CFD) framework that uses a combination of of Arbitrary Langrangian Eulerian (ALE) Dynamics and Adaptive Mesh Refinement (AMR). We describe updates and on-going work on the framework that allow for build portability on generic HPC (High Performance Computing) platforms. We also describe some of the more advanced algorithms that are available in the framework such as those which model surface tension effects in two and three dimensions. We introduce a new method for curvature and normal vector calculation in 2D, which we call the method of osculating circles. We benchmark this method and compare with other other volume of fluid (VOF) approaches to simulating surface tension effects. We predict how these algorithms will scale on these latest platforms such as the new Perlmutter system at NERSC which is a HPE Cray EX supercomputer with both GPU-accelerated and CPU-only nodes. We discuss the application of surface tension models to the interaction of a hydrogen droplet heated by an x-ray free electron laser with another hydrogen droplet.

Keywords: Numerical Algorithms, Computational Fluid Dynamics, Surface Tension, Mesh Adaptation, ALE Methods.

1 Introduction

Our CFD framework, known as PISALE (Pacific Island Structured-AMR with ALE) developed by the University of Hawaiʻi [1] (see also https://pisale.bitbucket.io/), uses a staggered-grid, Lagrangian formulation with position and velocity being nodal variables and density, internal energy, temperature, pressure, strain, and stress being zonal (cell centered) variables[1]. In this Lagrangian formulation (in vector and indicial notation $i, j, k = 1, 2, 3$) we have:

$$\frac{D\rho}{Dt} = -\rho\nabla\cdot\vec{U} = \rho U_{i,i} \tag{1}$$

$$\frac{D\vec{U}}{Dt} = \frac{1}{\rho}(\nabla\cdot\boldsymbol{\sigma} + B) = \frac{1}{\rho}(\sigma_{ij,j} + B_i) \tag{2}$$

$$\frac{De}{Dt} = \frac{1}{\rho}V\boldsymbol{s}:\dot{\boldsymbol{\varepsilon}} - P\dot{V} = \frac{1}{\rho}V\left(s_{ij}\dot{\varepsilon}_{ij}\right) - P\dot{V} \tag{3}$$

where

$$\frac{D}{Dt} = \frac{\partial}{\partial t} + \vec{U}\cdot\nabla$$

is the substantial derivative, ρ is the density, $\vec{U} = (u, v, w)$ is the material velocity, B is the body force per unit volume, t is time, $\boldsymbol{\sigma}$ is the total stress tensor, P is the pressure, e is the internal energy, V is the relative volume ($\rho V = \rho_0$ where ρ_0 is the reference density), $\boldsymbol{s}$ is the deviatoric stress defined as,

$$s_{ij} = \sigma_{ij} + P\delta_{ij} \tag{4}$$

where δ is the Kronecker delta, and $\dot{\boldsymbol{\varepsilon}}$ is the strain rate tensor defined as

$$\dot{\varepsilon}_{ij} = \frac{1}{2}\left(\frac{\partial U_i}{\partial x_j} + \frac{\partial U_j}{\partial x_i}\right) \tag{5}$$

A Volume Of Fluid (VOF) formulation with interface reconstruction is used to represent multiple materials and handle material advection during the ALE remap step. During a simulation, the volume fraction of each material in a mixed zone is stored and the actual interface is only reconstructed as needed, such as during mesh refinement. Volume fractions in mixed zones are used for weighting the pressures, densities, stresses, etc., to obtain cell averaged quantities. Volume fractions are used during refinement of neighboring zones to determine the orientation of each interface and the actual location is determined by the volume fraction in the zone. A 2nd-order predictor-corrector model is utilized for time integration. We mix the ALE scheme with adaptive mesh refinement (AMR) and in the code both the ALE and the AMR schemes can work independently to give zonal modification and potentially improve accuracy and/or runtime to solution.

Surface tension models within the PISALE framework have been implemented in a several ways[2]. These models were originally introduced to allow for simulation of phenomena such as droplet deformation in EUV generation [3], and include the ability to deal with multiple material interfaces unlike more restrictive methods based on level sets [4]. One method is implemented by adding a third-order space derivative to the stress that is derived thermodynamically based on the Van der Waals-Cahn-Hilliard free energy. A second approach is based on two-fluid VOF models, focusing on an implementation which calculates the curvature based on the height function approach. Although this is a two fluid model, PISALE uses a version that converts a single material density into the volume of fluid methodology and utilizes the same algorithm for a single-fluid implementation and the method is extended to three dimensions. We describe the behavior of these methods in the context of accuracy, efficiency, and parallel scaleablility.

2 PISALE SAMRAI update, build update

One of the long-term goals for the PISALE codebase is to make the code portable to new platforms and architectures. Most of the abstraction over basic MPI protocols is handled by the Scalable Structured-AMR Application Infrastructure (SAMRAI) library, which has continuously improved scaling performance of the underlying variable tracking primitives[5]. The recently released SAMRAI version 4.0 added support for the RAJA framework [6], which, if incorporated into the PISALE framework, would allow us to rewrite several inner loops to run as GPU parallel loops within MPI nodes. The upgrade from version 3.3 (the version currently used by PISALE) to version 3.4 is a major undertaking, requiring an update to almost every C++ file in the codebase, and is not yet complete.

Additionally, to enable rapid development, a new build system was created. Previously, PISALE build scripts were written primarily for certain machines (such as the DOE's NERSC Cori HPC system), and required significant effort to port to additional systems. With the addition of a traditional configure script and a significant reworking of the makefile scripts, we now have the capability to quickly and easily port PISALE to new systems, including large systems such as NSF's TACC Frontera HPC system and smaller ones such as the University of Hawaii's and University of Minnesota's clusters. The underlying tools (GNU make and POSIX sh) are available on practically any UNIX-like platform, so we believe that PISALE can now be compiled on any platform where the dependencies are supported. We also now have the ability to automatically do "dirty builds", that is, builds where only a few necessary object files are recompiled, reducing compilation times significantly. These capabilities have already proven useful in several projects using PISALE, including this work and the work of Yip et al. [7] appearing in these proceedings.

We expect that, as the PISALE codebase continues to evolve and is upgraded to make use of GPU computers such as the NERSC Perlmutter system, the significant time and effort which went into producing this new build system will prove to have been a worthwhile investment.

3 Surface Tension Models

In fluid dynamics theory, the surface tension between two fluids is typically understood as a boundary condition on the two stress tensors of the different fluid regions:

$$n \cdot \sigma - n \cdot \hat{\sigma} = \gamma n(\nabla \cdot n) - \nabla \gamma$$

Where γ is the surface tension coefficient, a scalar function defined on the boundary which depends on the physical characteristics of both fluids, n is the inward unit normal vector of the boundary, σ is the total stress tensor of fluid 1 and $\hat{\sigma}$ is the total stress tensor of fluid 2, both evaluated at the boundary. γ is typically approximated with a constant, which simplifies the equation to the Young-Laplace equation,

$$n \cdot \sigma \cdot n - n \cdot \hat{\sigma} \cdot n = \gamma(\nabla \cdot n)$$

Temperature-dependent models are sometimes relevant, in which case the tangential component of surface tension is called the thermo-capillary force [8]. The surface tension coefficient can also depend on other properties of both fluids in a neighborhood of the boundary, leading to other capillary forces. In this paper, only the case of constant surface tension coefficient is considered. In the future, we plan to implement surface tension as a function of the dynamically evolving field variables such as temperature and density.

The surface tension stress imbalance can be interpreted as a body force per unit area F_{st} acting perpendicular to the boundary, which is dependent on the curvature of the boundary. For some applications, intricate algorithms have been developed to track the evolving topology of the boundary [9]. However, in Eulerian and Arbitrary Lagrangian-Eulerian (ALE) codes such as PISALE, this approach is often impractical. Instead the typical approach, introduced by Brackbill et al.[10], is to track a relative volume-of-fluid density $\tilde{c}$ and use it to approximate a smooth surface tension body force. This is referred to as the Continuum Surface Force (CSF) model. A common method of curvature calculation that avoids large parasitic currents is to use a height function to calculate curvature, and to calculate gradient of $\tilde{c}$ for the normal vector.

In PISALE, multiple distinct methods have been developed for both 2D and 3D[2]. All of the models are based on the split-operator paradigm, where a module updates the velocity or internal energy field based on the geometry of the boundary after each complete hydrodynamic time step. The boundary is not explicitly tracked, instead the VOF field $\tilde{c}$ is tracked. In essence surface tension effects are calculated separately from the rest of the framework. This is efficient, but requires care to avoid creating an excessively large velocities which surpasses the Courant limit. There are no fundamental restrictions to using multiple materials with surface tension in the same simulation, however there are technical challenges when choosing the surface tension coefficient between two materials, and in detecting where and what can be considered a boundary. Currently, we assume a cell is a boundary cell only if it sits between two cells of different type (mixed or one of potentially several pure materials), and the surface tension coefficient of a material is constant and does not depend on what material lies on the other side of a boundary. When two fluids which both have surface tension touch, both of their surface tension forces are evaluated and added together. This treatment may not be accurate if surface tension effects are required when the boundary is truly diffuse (such as mixed vapors) and in the case of pathological boundary geometries, such as droplets with radius below the mesh resolution.

Previous work also included studies using an energy-preserving discretization of a third-order stress term which is derived thermodynamically based on the Van der Waals-Cahn-Hilliard free energy which we refer to as the Korteweig-type method. This method is not used for any current applications of PISALE due to the excessively fine mesh which is required to resolve the boundary, however, it is capable of handling temperature-dependent surface tension and suppresses parasitic currents[2], and has a fourth-order time integration scheme which would be difficult to replicate in models which require the curvature of the boundary to be calculated.

In all of the models in use, surface tension force is considered to be a cell-centered body force, which is then averaged into a node-centered body force term and added into the velocity field after each time step for first-order time integration. Motion of the boundary is implied during material advection. We plan on implementing second-order time integration by directly inserting F_{st} into the body force term of the core Lagrangian solver, but implementation difficulties have so far prevented this. Note that in PISALE, pressure is not a stored variable, instead it is calculated from density and internal energy (or temperature) using a tabular EOS when needed. Therefore it is not obvious a priori that the Young-Laplace equation would be satisfied or even approximated by our surface tension models, especially since they imply a discontinuity in pressure. Instead the velocity and motion of the boundary which would be caused by that same pressure imbalance are approximated, and in practice a pressure imbalance proportional to surface tension force is generated by the velocity field, even in the situation of a droplet suspended in vacuum.

3.1 2D: Differentiating the Height Function

One method stems from directly differentiating the height function values using central differences to evaluate the equations

$$\nabla \cdot n = \kappa = \frac{-h''(x)}{(1+(h'(x))^2)^{\frac{3}{2}}}$$

$$n = \frac{\nabla \tilde{c}}{|\nabla \tilde{c}|}$$

This method is simple and effective on modestly distorted meshes, and its properties have been well-studied by other authors [9, 11, 8, 12]. In our implementation, we assume that $h(x) = \frac{\partial y}{\partial i} i(x,y) + \frac{\partial y}{\partial j} H(i(x,y))$, where H is the discrete height function found by summing $\tilde{c}$ along columns in a 3x5 stencil. $\frac{\partial y}{\partial i}$, $\frac{\partial i}{\partial x}$ and $\frac{\partial y}{\partial j}$ are assumed to be constant and are estimated from the geometry of the mesh. In the best case scenario of a regular linear mesh and an exact height function, this method provably achieves $O(\Delta x^2)$ convergence, but cannot be applied to very irregular or nonlinear meshes without significant modification, because errors in the second derivative of H blow up as $\Delta x \to 0$.

3.2 2D: Osculating Circles

Another method, which we believe to be novel, instead uses the height function as a heuristic for approximating the osculating circle of the boundary. The osculating circle of a curve at a point is defined to be the circle which best fits the curve at that point, or which is the largest circle that rolls along both sides of the curve at that point. Such a circle has the same radius of curvature as the curve [13]. If $E(t)$ is defined to be the center of the osculating circle of the curve c at time t (E is called the evolute of c), then the inward normal vector of c at t is $\frac{E(t)-c(t)}{|E(t)-c(t)|}$ and the curvature of c at t is $\frac{1}{|E(t)-c(t)|}$.

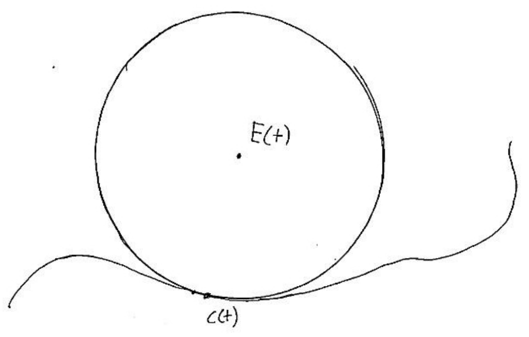

Figure 1: An example of an osculating circle.

The osculating circle for a curve can be approximated by choosing three points that lie approximately on the curve and solving a linear system of equations. The curvature and normal vector calculations in the equation $F_{st} = \gamma\kappa n$ are combined into a compact form.

Osculating circles are sometimes used for curvature estimation in the field of discrete differential geometry[14][15], although not to our knowledge using exactly the same method, and usually in different contexts. Mathematical proofs of convergence are complicated by the fact that one of the key terms in our circle construction is on the order of $\Omega(\Delta x^{-1})$. However, extensive experiments using random functions indicated the convergence estimates in figure 6, which show that the normal vector always converges as $O(\Delta x)$, and for regular meshes both normal vector and curvature converge as $O(\Delta x^2)$. Most interesting are the curvature estimates for irregular meshes, which have relative error converging to a small but nonzero value quickly on the order of $O(\Delta x)$. The same steady error values emerged from all tested functions in all function classes (including some which were not included in the chart for brevity), leading us to believe they are constants of the method rather than of any particular function. We dub these "ratio constants" and denote them $\beta(r)$, where r is the ratio between the larger step and the smaller step. Here are some approximate values: $\beta(2) \approx 2/30$, $\beta(1.5) \approx 1/27$, $\beta(1.1) \approx 1/123$, $\beta(1.001) \approx 8.33125e-5$. We suspect these ratio constants may be due to systemic biases introduced by the consistently irregular point choices. In all of our tests with a regular orthogonal mesh, with or without an exact height function, the osculating circle method had nearly identical curvature values to the method of calculating first and second derivatives with central differences discussed in the previous section.

Besides significantly more accurate normal vector determination which converges better than $\nabla\tilde{c}$, the advantage of osculating circles is that it imposes only very weak assumptions on the mesh geometry to get accurate values. In our tests (see figure 5 at the end of this document), while both methods for curvature calculation suffered in the presence of a non-conformal, non-linear mesh, the normal vectors using the osculating circle method were visibly more accurate than the normal vectors obtained by taking the gradient of $\tilde{c}$ (with no correction for linear mesh skew). In the case of a non-linear conformal mesh, the method of osculating circles had significantly better curvature values and again visibly better normal vectors, although both methods still suffered from inaccuracies in $\tilde{c}$. Therefore we believe this method is an upgrade to the more traditional height function methods, as it makes the use of conformal and non-regular meshes feasible, in particular mesh adaption would probably not degrade the performance of the surface tension code. It is possible that a better choice of heuristic could achieve better results as well.

Unfortunately, there is no natural generalization of this method to three dimensions. The osculating circle of a curve is useful in this context because its curvature and normal vector can be calculated without an explicit parameterization, it can be approximated very quickly from a few sample points, and it captures all first- and second-order information about the curve. Future work may focus on finding a class of surfaces with similar properties for use in three-dimensional surface tension models, or using an iterative method to find lines of curvature for the boundary and measuring their curvature using osculating circles.

One thing that was noted while studying the 2D models in PISALE was extreme sensitivity to inaccuracies in the underlying volume-of-fluid field, that is, if the height function is not exactly equal to the area under the true boundary (which in test cases can be known precisely) then there are significant errors in curvature, typically manifesting as alternating under- and over-estimates of curvature corresponding to a piecewise-linear boundary. It is easy to show that if the boundary is truncated to only first-order terms of the Taylor approximation within each column of the height function, the height function will perfectly find points on the boundary. In practice, only the boundary within each cell is approximated with a straight line, and there is no guarantee that the lines will match up or be consistent with a Taylor approximation. Thus even when more than one point is used per boundary cell in a polygonal approximation of a circle, it is possible to get significant errors in the height function. Errors of this kind prevented our practical tests from showing second-order convergence in figure 2.

A test case we ran for the 2D models was a 666.6666 nm-wide circular droplet of liquid aluminum at 2000K, surrounded by vacuum. Aluminum has a very accurate EOS in our framework, meaning that it offers the most fidelity for calculating the equilibrium needed in these tests. The material properties of aluminum were not relevant and were not simulated, in particular there was no viscous damping at all. The surface tension coefficient was set to 10^9 N/m to exaggerate the effects of surface tension. The mesh had cells/radius of 90 and a VOF accuracy was 2 points/boundary cell. The droplet developed parasitic flows around 10^{-4} times the magnitude of surface tension forces. Unlike in the case of a viscous droplet investigated by some other authors, the flow field did not stabilize over time, instead it evolved significantly as the surface tension forces compensated for the bulging caused by parasitic currents. We did not consider large velocity

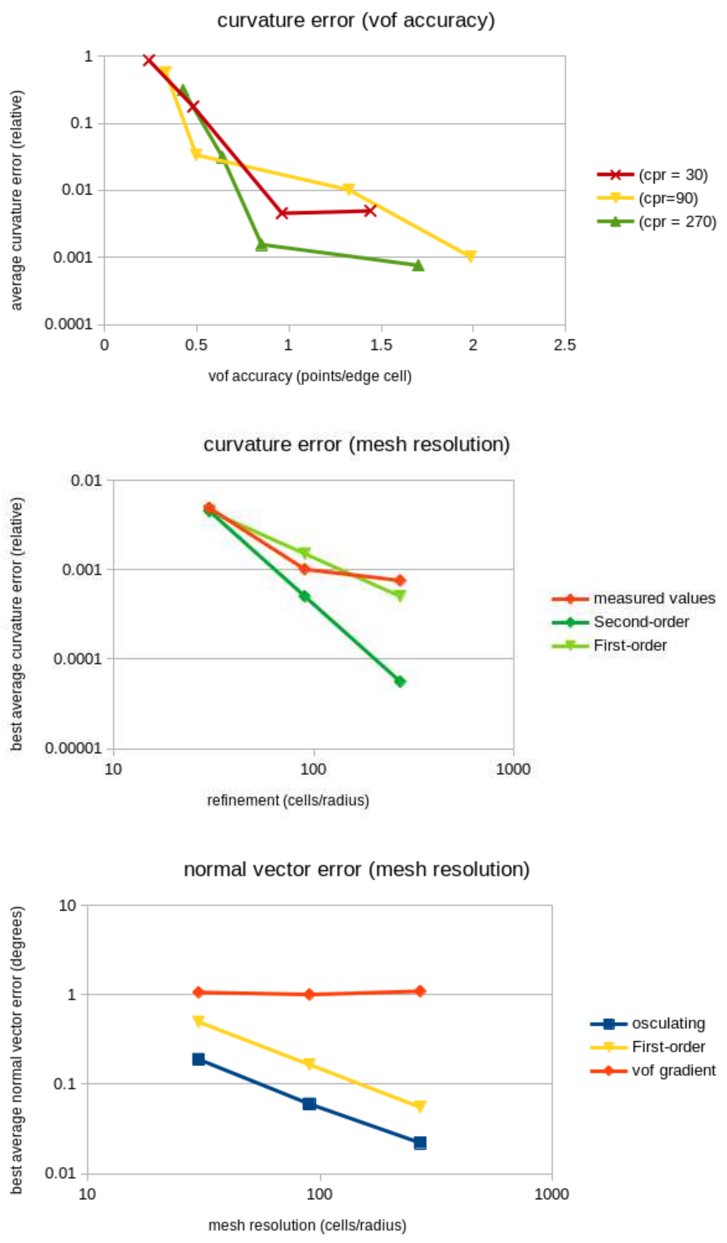

Figure 2: **Top**: Plot of relative error in the estimated curvature of a circle measured at different refinement levels (cpr = cells per radius) for both 2D curvature methods, measured on an approximation of a circle at time T=0, showing a clear exponential relationship between VOF accuracy and curvature accuracy. VOF accuracy is measured using the number of points in a regular polygon approximating a circle.
Middle: Plot of the best-measured relative error for different mesh refinement levels. Defects in the underlying VOF field prevented second-order convergence.
Bottom: Plot of relative error in the estimated normal vector of a circle for the method using osculating circles ("osculating") and taking the gradient of the volume-of-fluid field $\tilde{c}$ ("vof gradient"). No significant relationship between VOF accuracy and error was found for either method.

"parasitic" when it was on the boundary of the droplet and pointing in a logical direction to correct for a bulge. The droplet's center of mass remained stable over the course of 209 ns, indicating no movement of

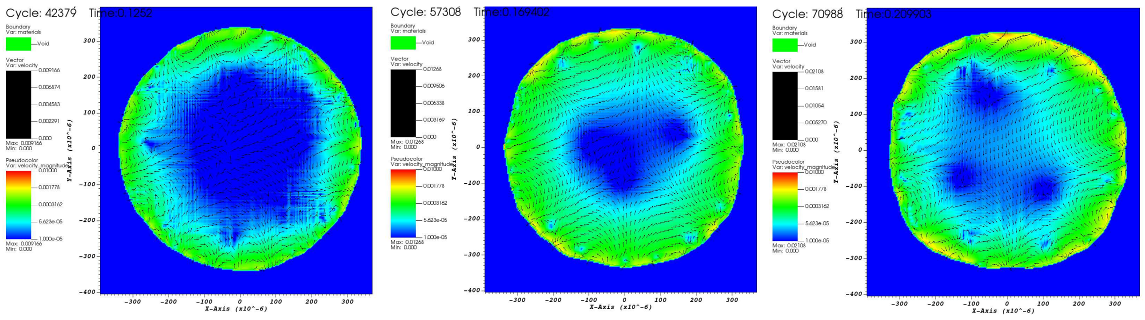
Figure 3: Pictures of parasitic currents and droplet deformation at different time points, colored with velocity magnitude in log scale. From left to right: t = 125 ns, t = 169 ns, t = 209 ns. This does not match the stable, organized flow characteristic of parasitic currents in viscous droplets.

the drop due to parasitic currents. At $t = 209$ ns the average distance from boundary points to the original boundary was approximately 5.779 nm. Similar tests made with coarser meshes showed similar results; in particular, refinement did not result in smaller parasitic flow, which is consistent with the predictions of Harvie et al.([16]). It is unclear if the boundary would ever stabilize, the surface tension coefficient may be too high for the oscillations in the boundary to eventually cancel out completely.

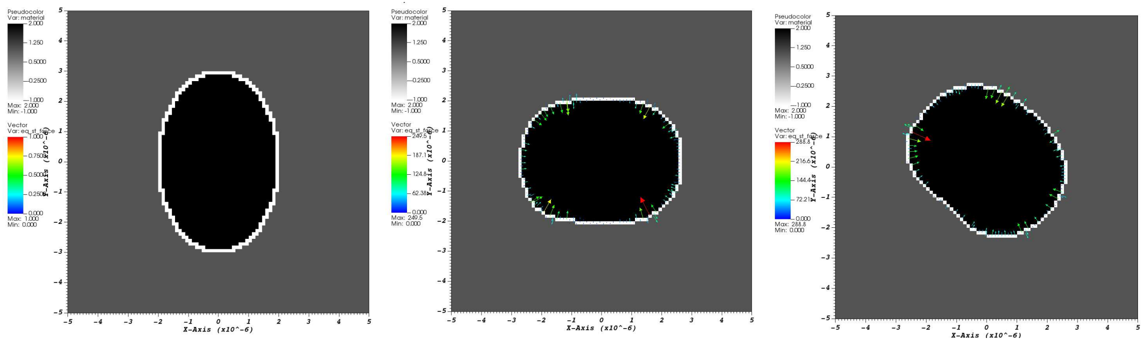
Figure 4: Pictures of an oscillating ellipse at different time points. From left to right: t = 0 ns, t = 2.1 ns, t = 12 ns. Arrows are surface tension forces acting on the surface of the droplet. No forces are present at t = 0 because surface tension forces are only calculated after the initial time step.

Another test we tried was to find the oscillation frequency of an ellipse-shaped droplet. The initial condition was a droplet of liquid aluminum at 2000K, in equilibrium with surrounding vacuum, with major axis of 3 nm and minor axis of 2 nm. The density of the aluminum was calculated using our EOS tables at 2389 kg/m^3. The surface tension was set to 100000 N/m. Material properties of aluminum were not simulated, so the aluminum was essentially treated as an incompressible non-viscous liquid.

The ellipse deformed into an ellipse-like shape with similar dimensions, but flipped about the line $x = y$, exactly once, forming one half-cycle which took approximately 2 ns. The new ellipse-like shape persisted for approximately 0.6 ns before starting to deform significantly again, becoming a vaguely circular blob which kept oscillating in an unstructured way until we stopped the simulation at $t = 12$ ns. The oscillation frequency in the second mode was therefore between 0.18 GHz and 0.25 GHz. This differs substantially from the prediction of 1.5 THz which can be computed from the classical equations of Rayleigh[17]. We suspect that this discrepancy is due to the fact that the oscillation had a large amplitude, meaning some nonlinear terms which are neglected in the derivation of the oscillation frequency are not negligible in this setting. This may also be what caused the droplet to lose its apparent ellipse shape. Another possibility is this is related to the two-dimensional approximation, since previous oscillation frequencies in three dimensions calculated by similar code models matched Rayleigh predictions for certain configurations[2]. In any case, more study is required.

3.3 3D: Polynomial Approximation

In 3D, we again use height function, this time evaluated on a 3x3x7 stencil. Six points are used for a linear regression to generate a 2nd-degree polynomial approximation $P(i,j) = P_0 i * i + P_1 i * j + P_2 j * j + P_3 * i + P_4 * j + P_5$ of the discrete height function $H(i,j)$. This is used to create a polynomial approximation $p(x,y)$ for $h(x,y)$ in a manner entirely analogous to that in subsection 3.1 above. The first and second derivatives of p at 0 can then be calculated to evaluate the curvature

$$\kappa = \frac{(1+(\frac{\partial p}{\partial x}(0))^2\frac{\partial^2 p}{\partial y^2}(0) - 2\frac{\partial p}{\partial x}(0)\frac{\partial p}{\partial y}(0)\frac{\partial^2 p}{\partial x \partial y}(0) + (1+(\frac{\partial p}{\partial y}(0))^2)\frac{\partial^2 p}{\partial x^2}(0)}{1+(\frac{\partial p}{\partial x}(0))^2+(\frac{\partial p}{\partial y}(0))^2)^{\frac{3}{2}}}$$

The normal direction is calculated using the gradient of $\tilde{c}$. A more detailed description and mesh refinement studies for this method can be found in [2], we only tested to confirm that spherical droplets remained stable over long time periods and that perturbed droplets returned to spherical shape, which is the relevant property of surface tension in the present application.

4 Efficiency of Surface Tension Methods

All of our height-function based surface tension methods are highly parallel, since they require only data that is already available on every AMR patch and produce no new data – only the velocity field is updated by the surface tension modules. When PISALE is upgraded to support the Raja library, they could also easily be turned into a stream computation by writing a GPU kernel to operate over each stencil in parallel, possibly at the expense of memory efficiency.

Our 2D implementations of surface tension are both very fast even without significant optimization. Typically the surface tension step takes a negligible amount of time compared to the rest of the step in PISALE's main loop. This may be because these implementations select the orientation before calculating surface tension. The 3D code, in contrast, calculates surface tension up to 6 times for each cell and picks the best value based on the magnitude of the derivative of the height function. Each calculation involves a linear regression, so the 3D surface tension code is substantially slower than the 2D code, though still fast enough that no significant optimization is needed for the code to be used.

None of the height function methods required an excessively fine mesh. In fact, in our tests the accuracy of the VOF field itself had a stronger impact on surface tension than mesh refinement did. This is in contrast to earlier attempts at using a Korteweg-type model for surface tension in PISALE [2], which required a very fine mesh to properly resolve the boundary, causing small time steps. The code also required several variables to be tracked across multiple AMR domains (which increases the necessary MPI communication using SAMRAI) and thus the parallel efficiency suffers slightly. Therefore while the Korteweg-type model has vastly improved accuracy and has a more physically robust description of the surface tension force than the CSF model, it is less useful for use in long-period simulations of transient behavior.

5 Application: Modeling hydrogen droplets hit with a laser

PISALE is currently being used to study the dynamics of cryogenic hydrogen droplets heated by an x-ray free electron laser (XFEL) and how the expanding droplet interacts with unheated droplets as described by Parisuana[18]. While the surface tension coefficient of cryogenic hydrogen is relatively small, the small size of the droplets (of order 5 microns) and corresponding large curvature can result in a significant restoring force to resist deformation of the unheated droplets. The initial simulations shown by Parisuana[18] do not include surface tension effects. By using the same equation of state for hydrogen with multiple labels, our two-fluid model could easily be extended for use in single-fluid scenarios such as a liquid hydrogen droplet surrounded by low-pressure hydrogen vapor.

The relevant property of surface tension in this scenario is that spherical droplets could be stable even when interacting with vapor from a heated droplet. Therefore the relevant features of our numerical surface tension model are that it does not develop enough parasitic currents to render the droplet unstable and that it does not have a non-spherical droplet shape as an equilibrium. This is a very important problem because the target droplets are in a line and if a heated expanding droplet deforms the following droplet, that droplet cannot be used. This reduces the effective repetition rate and corresponding data collection rate of the experimental system.

The surface tension coefficient is measured in units of work/change in area, meaning, it represents how much work is required to change the surface area of a droplet, ignoring the fluid's momentum. Using the surface tension coefficient of $0.003\text{J}/\text{m}^2$ which can be estimated from standard equations [19], and a rough approximation of the final first droplet shape as a truncated sphere, we approximate the required extra work to match the observed deformations in the first droplet at t=9 ns to be approximately 6×10^{-15} J. It is obvious that much more than this amount of energy had already been deposited in the droplet by that time, so surface tension probably would not change the outcome significantly for the first droplet. It remains to be seen how much energy would be deposited in the second droplet and how much deformation it could effect.

6 Conclusion and Future Work

In this paper, we describe several upgrades to the code and build system of the CFD framework PISALE, enabling the use of its combination of structured adaptive mesh refinement (AMR) and arbitrary lagrangian-eulerian (ALE) methods for new computational architectures. We also detail the surface tension models in PISALE and describe a potential new method of curvature calculation, which we call the osculating circle method, that is potentially more flexible and robust in the presence of mesh adaption than typical height function methods while preserving most of their positive features. While our initial findings are promising, we believe a deeper study of the osculating circle method and its possible application to 3D is warranted. We also conclude that our existing 3D surface tension code is applicable to a new application area aimed at modeling the stability of cryogenic hydrogen droplets in free-electron laser systems such as the SLAC.

7 Acknowledgements

We would like to thank many additional members of the PISALE team for their expertise and assistance throughout all aspects of our study and for their help in writing this manuscript. The PISALE code is supported by the National Science Foundation, under Office of Advanced Cyber Infrastructure Award Number 2005259, the U.S. Department of Energy, Office of Science, under Fusion Energy Sciences Research Division Award Number DE-SC0021374, and the Office of Naval Research, under ONR MURI Award Number N00014- 20-1-2682. This research used resources of the National Energy Research Scientific Computing Center (NERSC), a U.S. Department of Energy Office of Science User Facility located at Lawrence Berkeley National Laboratory, operated under Contract No. DE-AC02-05CH11231 using NERSC awards ERCAP0020882 and ERCAP0020890. The technical support and advanced computing resources from University of Hawaii Information Technology Services – Cyberinfrastructure, funded in part by the National Science Foundation MRI award Number 1920304, are gratefully acknowledged.

References

[1] Alice Koniges, Nathan Masters, Aaron Fisher, David Eder, Wangyi Liu, Robert Anderson, David Benson, and Andrea Bertozzi. Multi-material ale with amr for modeling hot plasmas and cold fragmenting materials. *Plasma Science and Technology*, 17(2):117, 2015.

[2] W. Liu, A. Koniges, K. Gott, and D. Eder. Surface tension models for a multi-material ale code with amr. *Int Computers and Fluids*, 151:91–101, 2017.

[3] Michael A. Purvis, Alexander Schafgans, Daniel J. W. Brown, Igor Fomenkov, Rob Rafac, Josh Brown, Yezheng Tao, Slava Rokitski, Mathew Abraham, Mike Vargas, Spencer Rich, Ted Taylor, David Brandt, Alberto Pirati, Aaron Fisher, Howard Scott, Alice Koniges, David Eder, Scott Wilks, Anthony Link, and Steven Langer. Advancements in predictive plasma formation modeling. In Eric M. Panning, editor, *Extreme Ultraviolet (EUV) Lithography VII*, volume 9776, pages 159 – 170. International Society for Optics and Photonics, SPIE, 2016.

[4] J. A. Sethian and Peter Smereka. Level set methods for fluid interfaces. *Annual Review of Fluid Mechanics*, 35(1):341–372, 2003.

[5] Brian T.N. Gunney and Robert W. Anderson. Advances in patch-based adaptive mesh refinement scalability. *Journal of Parallel and Distributed Computing*, 89:65–84, 2016.

[6] D. A. Beckingsale, J. Burmark, R. Hornung, H. Jones, W. Killian, A. J. Kunen, O. Pearce, P. Robinson, B. S. Ryujin, and T. R. W. Scogland. RAJA: Portable Performance for Large-Scale Scientific Applications. *IEEE/ACM International Workshop on Performance, Portability and Productivity in HPC (P3HPC)*, pages 71–81, 2019.

[7] Peter T. Yip, Erik Torres, Ioannis Nompelis, Thomas E. Schwartzentruber, Austin J. Andrews, Devin AJ McGee, Ioannis Pothos, Nathan A. Bellefeuille, Christopher J. Hogan, and David C. Eder. Arbitrary lagrangian eulerian simulations of high speed particle impacts encountered during hypersonic flight. *Eleventh International Conference on Computational Fluid Dynamics (ICCFD11)*, 2022.

[8] V. Nilsson. Modeling of Temperature-Dependent Surface Tension Forces. Master's thesis, Chalmers University of Technology, 2019. https://hdl.handle.net/20.500.12380/256854.

[9] Daniel Lorstad, Marianne Francois, and Wei Shyy Laszlo Fuchs. Assessment of volume of Fluid and immersed boundary methods for droplet computations. *Int J Numerical Methods in FLuids*, 46:109–125, 2004.

[10] J U. Brackbill, D. B. Kothe, and C. A. Zemach. Continuum method for modeling surface tension. *J Comput Phys*, 100:335–354, 1992.

[11] S. J. Cummins, M. M. Francois, and D. B. Kothe. Estimating curvature from volume fractions. *Comput Struct*, 83:425–434, 2005.

[12] Z. Guo, D. F. Fletcher, and B. S. Haynes. Implementation of a Height Function Method to Alleviuate Spurious Currents in CFD Modelling of Annular Flow in Microchannels. *Applied Math Modelling*, 39:4665–4686, 2015.

[13] M. Spivak. *A Comprehensive Introduction to Differential Geometry*, volume 2. Publish or Perish inc., 1999.

[14] M. Worring and A. W. M. Smeulders. Digital Curvature Estimation. *CVGIP: Image Understanding*, 58:466–382, 1993.

[15] D. Coeurjolly, S. Miguet, and L. Tougne. Discrete Curvature Based on Osculating Circle Estimation. *IWVF4*, pages 303–312, 2001.

[16] D.J.E.Harvie, M.R.Davidson, and M.Rudman. An analysis of parasitic current generation in Volume of Fluid simulations. *App Math Modelling*, 30:1056–1066, 2005.

[17] A. Aalilija, Ch.-A. Gandin, and Elie Hachem. On the analytical and numerical simulation of an oscillating drop in zero-gravity. *Computers and Fluids, Elsevier*, 197:104362, 2020.

[18] C. Parisuana Barranca, D. C. Eder, M. Gauthier, C. Schoenwaelder, C. A. Stan, and S. H. Glenzer. Cfd modeling of droplets heated by an x-ray free electron laser. *Eleventh International Conference on Computational Fluid Dynamics (ICCFD11)*, 2022.

[19] A. Muleroa and I. Cachadiña. Recommended Correlations for the Surface Tension of Common Fluids. *J. Physical and Chemical Reference Data*, 41:043105, 2012.

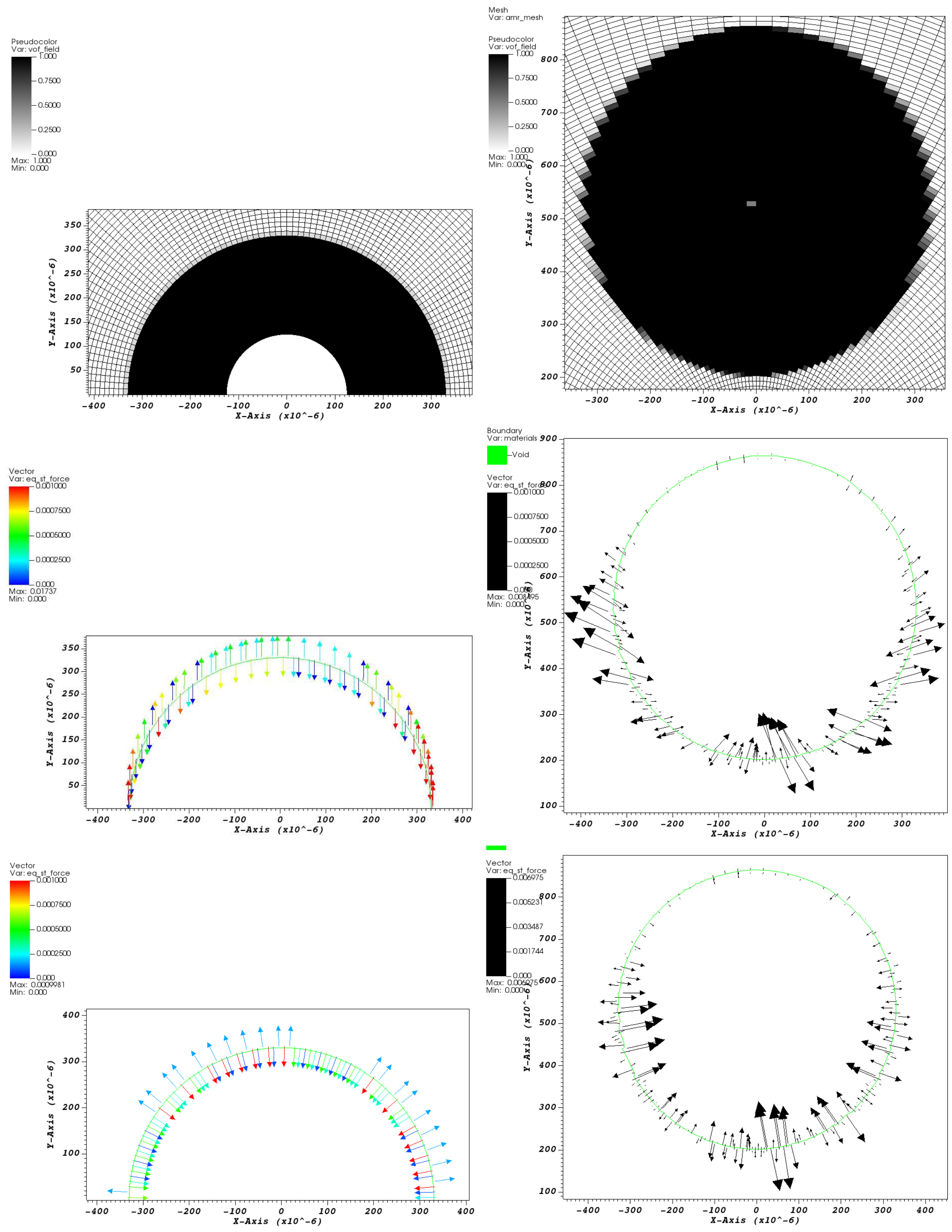

Figure 5: **Top Row**: simulation setup for conformal mesh tests (left) and skewed mesh tests (right).
Middle Row: Calculated surface tension forces using the DHF+VOF method, with conformal mesh (left) and skewed mesh (right). The plot on the left uses constant vector lengths so the normal vectors are more apparent. Average curvature error on the left was 1.2204, on the right was 1.5547.
Bottom Row: Calculated surface tension forces using the osculating circle method, with conformal mesh (left) and skewed mesh (right). The plot on the left uses constant vector lengths to match the above plot. Average curvature error on the left was 0.4809, on the right was 1.192

ratio $\frac{\Delta x_1}{\Delta x_2}$	est. convergence exponent of normal vector	est. convergence exponent of curvature (relative)	curvature error steady value (relative)
random polynomials $a_4x^4 + a_3x^3 + a_2x^2 + a_1x$			
2	1	0	0.066
1.5	1	0	0.037
1.1	0.998	0	0.008
1.001	1.22	0.032	8e-5
1	2	2	0
random skew-ellipses $y = a_4 sqrt(a_3^2 - a_2^2x^2 + a_1^2x)$			
2	1.022	0.012	0.066
1.5	1.02	0.01	0.037
1.1	1.025	0.006	0.008
1.001	1.175	0.069	8e-5
1	2	1.737	0
random circles $y = a_4 sqrt(1 - x^2)$			
2	1	0.003	0.066
1.5	1	0.002	0.037
1.1	1	0.003	0.008
1.001	1	0.051	8e-5
1	2.27	2	0
random sinudoids $y = a_4 sin(a_3x) + a_2 cos(a_1x)$			
2	1	0	0.066
1.5	1	0	0.037
1.1	0.992	0	0.008
1.001	1.395	0	8e-5
1	2	2	0
random polynomials, no HF integral			
2	2	1	0
1.5	2	1	0
1.1			
1.001			
1	2	2	0

Figure 6: Estimates of convergence exponents and asymptotic error of osculating circle normal vectors and curvature values, for different families of randomly chosen functions. Most of the decimals in the exponents are byproducts of measurement error (in particular most of the "convergence" in the curvature values is really convergence to an incorrect number) and numerical instability. Many numbers are rounded for clarity.