
Proceedings Paper:
Dennison, R. and Maddock, S. (2026) Extending implicit density projection for multiphase fluids. In: Li, P., Ma, l., Wan, L., Sheng, B., Kim, J., Thalmann, D. and Magnenat-Thalmann, N., (eds.) Advances in Computer Graphics: 42nd Computer Graphics International Conference, CGI 2025, Hong Kong, China, July 14–18, 2025, Proceedings, Part III. 42nd Computer Graphics International Conference, CGI 2025, 14-18 Jul 2025, Hong Kong, China. Lecture Notes in Computer Science (LNCS), LNCS 16508. Springer Cham, pp. 457-473. ISBN: 9783032222664. ISSN: 0302-9743. EISSN: 1611-3349.

https://doi.org/10.1007/978-3-032-22264-0_36

Extending Implicit Density Projection for Multiphase Fluids

Robert Dennison[0009−0007−0481−0480] and Steve Maddock[0000−0003−3179−0263]

The University of Sheffield, UK
{rtdennison1,s.maddock}@sheffield.ac.uk

Abstract. To increase the realism of fluid simulations, interactions between different fluids (such as water and air, or water and oil) must be considered. This allows for the simulation of phenomena such as the 'glugging' effect seen when a filled bottle is turned upside-down and air must fill the space left by the liquid. A common approach to fluid animation is to use Particle-in-Cell (PIC) methods. However, such methods are known to lose fluid volume over time due to accumulating numerical errors, an issue that is exacerbated when simulating multiple interacting fluids.
Implicit Density Projection (IDP) is used to overcome volume preservation issues for PIC methods. However, it is formulated for single fluids. To address this, we present two novel extensions to IDP: generalised IDP and D1-IDP. Generalised IDP extends IDP to multiple fluids. D1-IDP then further improves on the volume preservation capabilities. We show that D1-IDP is particularly good in multiphase fluid simulations with complex fluid interfaces. We also demonstrate its applicability to multiphase simulations involving variable density fluids. D1-IDP is able to achieve a maximum volume error of $< 1.0\%$ for the majority of presented scenarios, while having a negligible impact on computation performance compared to generalised IDP. The code used to generate the results presented in this paper can be found at https://github.com/robden820/Multiphase_Fluids.

Keywords: Physical Simulation · Fluid Simulation

1 Introduction

A popular technique for simulating single-phase fluids is the Particle-in-Cell (PIC) method, which is used in several commercial systems, e.g. Blender and SideFx's Houdini. PIC combines the ease of advection of particle-based simulation techniques with the ease of pressure calculations of grid-based techniques. However, one issue with the technique is that simulated fluids appear to lose volume over the duration of the simulation. This is because although the underlying mathematical model of a PIC simulation treats the fluid as incompressible, numerical discretization errors cause particles to become more densely clustered over time, resulting in volume loss. Since the pressure solver is unaware of fluid density, once the fluid has lost volume due to particle clumping (and so increased

in density) it can't be rectified. It is therefore necessary to include some additional means of preserving the fluid volume. Implicit Density Projection (IDP) [16] is a technique to overcome this by tracking the density of fluid. The pressure solver then takes density into account as well as velocity, and so the fluid can maintain its initial volume. Although IDP has been shown to be very effective at preserving fluid volume for the duration of a simulation, it has yet to be adapted to multiphase simulations.

There is little research into the field of multiphase PIC simulations. The work of Boyd and Bridson [2] was the first to adapt PIC to simulate multiphase fluids, however, the proposed method of bumping particles away from the fluid interface increases particle clustering, exacerbating the issue of volume loss. To mitigate volume loss issues, they applied the proportional control method of Kim et al. [14], however this approach requires careful tuning of additional parameters and can lead to artifacts where fluid volume can fluctuate over the course of a simulation. More recently, Lyu et al. [20] proposed a multiphase narrowband FLIP model to reduce the computational cost of MultiFLIP, and Lyu et al. [19] proposed MultiAPIC, adapting the affine PIC method to multiphase fluid simulations. However, neither of these works address the issue of volume loss in multiphase fluid simulations using PIC.

Our paper improves on this prior work by making the following contributions:

- We present an extension of IDP called generalised IDP (gIDP), which allows IDP to be applied to multiphase fluids simulated using Particle-in-Cell methods. IDP has previously been demonstrated to be highly effective in single-phase simulations [16] but has not yet been adapted to multiphase fluid simulation.
- We propose a new approach called D1 Implicit Density Projection (D1-IDP). This is an improvement over gIDP, reducing volume loss whilst requiring fewer solver iterations, at a small increase in computational cost. The technique is particularly effective when complex fluid interfaces form.
- We demonstrate the improvements of D1-IDP on a range of multiphase fluid scenarios: mixing air and water to produce a 'glugging' effect, a Rayleigh-Taylor instability simulation involving oil and water, and a simulation involving fluids that variable density over time due to temperature change.

The remainder of this report is split into five sections. Section 2 covers related literature. Section 3 describes the details of adapting IDP to multiphase fluid simulation for Generalised IDP (gIDP), as well as introducing D1-IDP. The results in Section 4 provide a comparison between gIDP and D1-IDP, followed by a discussion in Section 5. Finally, conclusions are presented in Section 6.

2 Related Work

Three main techniques are used in the the fluid animation literature: Smoothed Particle Hydrodynamics (SPH), Particle-in-Cell (PIC) methods, and the Lattice Boltzmann Method (LBM). SPH is a purely Lagrangian approach, making

use of particles to discretize the fluid ([15], [12]). A particle-based approach allows for ease of advection, but is more challenging when it comes to enforcing incompressibility. LBM is an Eulerian simulation method, using a grid-based discretization which simplifies the pressure calculations at the cost of ease of advection ([4], [17], [18]). PIC methods are a hybrid approach, using multiple interpolation stages to transfer information between the particles and the grid ([6], [22]). While PIC methods benefit from the positives of both Lagrangian and Eulerian approaches, the interpolation stages result in numerical dissipation leading to fluids appearing more viscous. Our work concentrates on PIC methods. For more details on the topic of fluid simulation for computer graphics, see the survey paper by Wang et al. [26].

Particle-in-Cell (PIC) methods were first introduced by Harlow et al [10] and have become an industry standard technique for fluid simulations. PIC simulations have been used to simulate a wide variety of materials, including both fluids and deformable solids. Brackbill and Ruppel [3] introduced the Fluid Implicit Particle (FLIP) method to reduce artificial viscosity, however, whilst FLIP reduces the dissipation of the standard PIC approach, it does so in an unstable, noisy manner. PIC and FLIP simulations were later adapted to incompressible fluid flow by Zhu and Bridson [27]. Jiang et al. [13] introduced the Affine PIC (APIC) method to overcome the noise introduced by FLIP by storing an additional affine transformation matrix on each particle that allowed for maintaining rotational velocities as well as linear velocity, resulting in smoother splashing effects compared to FLIP. Following this, Fei et al. [5] extended APIC to introduce Affine FLIP (AFLIP), further increasing the dynamism of APIC. Fu et al. [6] then developed the Polynomial PIC (PolyPIC) method which generalised the affine matrix of APIC to higher order polynomials, which further reduced numerical dissipation and improved the preservation of angular momentum. Recently, Sancho et al. [22] introduced the Impulse PIC (IPIC) method which uses an impulse-based flow map to correct particle velocities to preserve vorticial details. Both PolyPIC and IPIC improve the preservation of smaller details in the simulation, allowing for more turbulent-appearing flows.

The trend in developments of PIC methods has been to reduce the numerical dissipation of the technique to allow for more dynamic and more realistic liquid behaviour. A different approach to improve simulation realism is to simulate multiphase fluids, to allow for effects such as liquid bubbling and 'glugging'. PIC methods were first adapted to allow for the simulation of multiphase fluids in the work of Boyd and Bridson [2], who introduced the MultiFLIP method. Lyu et al. [20] reduced the computational cost of MultiFLIP by using an adaptive narrowband simulation, only using FLIP particles around the fluid interface. Lyu et al. [19] then introduced MultiAPIC, which used APIC in the place of FLIP for multiphase simulation.

As previously mentioned, PIC methods suffer from volume loss over time, especially as more dynamic scenarios are simulated or larger timesteps are used. Multiple techniques have been proposed in the literature to overcome this issue in relation to single-phase fluid simulation. Kim et al. [14] first proposed solving

the pressure Poisson equation (PPE) for non-zero divergence, and was used to improve the volume preservation of MultiFLIP [2]. They introduced the proportional controller, which would adjust the divergence of a cell based on if the cell density was too low or too high. Ando et al. [1] introduced Smoothed Particle Hydrodynamics (SPH) spring forces to enforce a uniform particle distribution. Um et al. [25] proposed the use of a secondary, high-resolution grid which could be used for adjusting particle positions based on particle density. Kugelstadt et al. [16] introduced Implicit Density Projection (IDP) which involves using a second Poisson equation to explicitly track simulation density. Qu et al. [21] then introduced PowerPIC, which adapted the work of de Goes et al. [7] to PIC simulations, making use of power diagrams to enforce incompressibility. Guo et al. [8] use additional implicit incompressible SPH (IISPH, [11]) particles to solve an additional low resolution PPE to adjust the positions of FLIP particles to preserve volume. However, none of these works aim to address the problem of volume loss for multiphase fluid simulation.

3 Methods

This work builds on MultiFLIP [2] by adapting IDP [16] to multiphase fluid simulations to improve the volume preservation characteristics. First, Subsection 3.1 covers IDP for single-phase fluid simulation. In Subsection 3.2, we present gIDP, an extension of IDP to allow for its use in multiphase fluid simulation. Subsection 3.3 then proposes D1-IDP, a new technique for volume preservation for multiphase fluids. Subsection 3.4 then covers special cases where D1-IDP and gIDP require additional handling. Finally Subsection 3.5 describes the simulation of fluids with varying density based on temperature change.

3.1 Implicit Density Projection

A single-phase fluid simulation using PIC consists of particles and a staggered Marker-and-Cell (MAC) grid. Particles carry information about their mass and velocity, and the grid acts as a scratchpad for pressure calculations to enforce incompressibility. PIC fluids suffer from volume loss over time as numerical errors cause particles to clump together, which can't then become separated. One approach to solving this issue for single phase fluids is Implicit Density Projection, which involves introducing a second Poisson equation (PE) to enforce constant density at each time step.

The pressure Poisson equation (PPE) to be solved, given as Equation 8 in [16], is

$$\frac{\Delta t}{\rho_0}\Delta^2 p = \Delta \cdot \mathbf{u}^* + \frac{1}{\Delta t}(1 - \frac{\rho^*(t)}{\rho_0}) \tag{1}$$

where $\mathbf{u}^*$ is the intermediate velocity field, $\rho^*(t)$ is the density field at time t, and ρ_0 is the initial fluid density that we want to maintain throughout the simulation duration. The first term, $\Delta \cdot \mathbf{u}^*$, is the divergence of the intermediate

velocity field, and is included in the standard pressure solve. The new term, $\frac{1}{\Delta t}(1 - \frac{\rho^*(t)}{\rho_0})$, is the scaled density error, and is included in the IDP PPE. In a single iteration, this single Poisson equation is split into two separate Poisson equations, which are solved separately:

$$\frac{\Delta t}{\rho_0}\Delta^2 p = \Delta \cdot \mathbf{u}^* \tag{2}$$

$$\frac{\Delta t}{\rho_0}\Delta^2 p = \frac{1}{\Delta t}(1 - \frac{\rho^*(t)}{\rho_0}) \tag{3}$$

To solve Equation 3 we need the target density, ρ_0, of every cell, which for a single-phase fluid is the density of the fluid being simulated. A single simulation step for PIC fluids is given in Algorithm 1.

3.2 Generalised Implicit Density Projection

When simulating the interactions between two different fluid phases, each particle is classed as belonging to one of the phases. The work of Boyd and Bridson [2] proposed that each of these particle types should interpolate to a separate velocity field on the grid. These separate velocity fields are then combined to calculate a single divergence value for each cell, so that only a single PPE needs to be solved for the entire simulation. The surface tension coefficient and fluid interface curvature is used to calculate the pressure jump at the fluid interface for the PPE. When calculating the interface curvature, a level set is generated from the particles of both phases. The positions of particles that are within a specified distance to the level set are "bumped" away from the level set along its normal, to ensure that it is smooth and to prevent particle mixing at the interface (see Figure 1). The particles then interpolate information back only from the corresponding velocity field.

To adapt IDP to multiphase simulations, we must first determine the target density of each cell. Similarly to the pressure PE, a single density PE can be used for both phases, rather than requiring solving separate linear systems for each phase. For a multiphase fluid, there is no fixed initial density ρ_0 as different cells contain different fractions of each phase. For cells that are marked as belonging to one of the two phases (A or B), the target density is the corresponding phase density (ρ_A or ρ_B). For the interface cells, the target density is calculated as

$$\rho_0 = \phi\rho_A + (1 - \phi)\rho_B \tag{4}$$

where ϕ is the fraction of the cell occupied by phase A.

However, we found that this resulted in "popping" artifacts throughout the fluid, similar to what was seen in the original IDP work when using a combined Poisson equation. We therefore took a similar approach to Boyd and Bridson [2] and set the scaled density error to 0 for interface cells, effectively meaning gIDP is not applied at the fluid interface. This resolved the artifact issue and resulted in effective volume conservation for the two fluids. The multiphase PIC simulation loop is given in Algorithm 1.

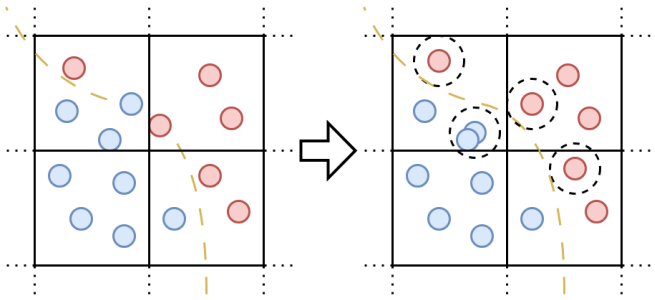

Fig. 1: A level set is generated from the positions of particles of both fluid phases (yellow). The positions of particles close to the interface are updated, moving them away from the level set along the level set normal vector. This ensures the interface is smooth and prevents particle mixing. As can be seen in the top left grid cell, this bumping stage is a cause of particle clustering. Based on Figure 5 in [2]

3.3 D1 Implicit Density Projection

The goal of IDP in is to enforce a constant density throughout the body of fluid. In PIC simulations the mass of particles is constant, as is the fluid volume represented by each particle. Since the mass and volume of particles are fixed, the result of enforcing constant fluid density is that an even distribution of particles is achieved throughout the fluid body. The idea behind D1-IDP is that since an even distribution of particles results in volume preservation, the physical fluid density can be disregarded. In the case of multiphase fluids, both fluid phases can be treated as having a density of 1.0 in the density Poisson equation. Unlike gIDP, we can apply D1-IDP across the fluid interface without causing artifacts. Equation 3 then becomes:

$$\frac{\Delta t}{n}\Delta^2 p = \frac{1}{\Delta t}(1 - \frac{n^*(t)}{n}) \tag{5}$$

where n is the desired number of particles in each grid cell, and $n^*(t)$ is the current number of particles in the grid cell. The number of particles is interpolated to the grid cells using Equation 6, rather than simply counting the number of particles in each grid cell:

$$n^*(t)_i = \sum_{p=0}^{P} w_{ip} \tag{6}$$

$$w_{ip} = N(\frac{x_p - x_i}{\Delta x}) \tag{7}$$

where x_p is the position of particle p, x_i is the position of grid cell i, Δx is the length of a grid cell edge. For this work we use linear interpolation:

Algorithm 1 A single iteration of a multiphase PIC simulation using IDP. Additional IDP steps are highlighted in teal, additional multiphase steps are highlighted in magenta.

Interpolate from particles to the grid
Calculate density at grid cell centres
Calculate cell density error
Solve density PE
Bump particles to adjust particle density
Calculate grid phi values and cell face fractions
Calculate interface curvature on the grid
Bump particles on the wrong side of the interface
Interpolate from particles to the grid
Apply external forces (e.g. gravitational forces)
Calculate grid cell divergence
Solve pressure PE
Update cell velocity from cell pressure
Extrapolate velocity field of each fluid phase.
Interpolate from the grid to particles
Advect particles

$$N(x) = \begin{cases} 1 - |x| & x < 1 \\ 0 & x \geq 1 \end{cases} \tag{8}$$

3.4 Degenerate Cases

When using IDP for single-phase fluids, the density error near the fluid surface is clamped to a minimum value of the fluid density, as these areas will not have an accurate density calculation due to the cells interpolating from a smaller number of particles. In a multiphase simulation using gIDP or D1-IDP this problem is avoided as the entire simulation domain contains an even distribution of particles. However, these degenerate cases can still occur at the domain boundaries and at solid boundaries, which must be considered (see Figure 2).

When using gIDP, if a cell has any neighbouring solid cells, the current uncorrected density, $\rho^*(t)$, is clamped to a minimum value of the cell's target density (Equation 4). For D1-IDP, the current number of particles in the grid cell, $n^*(t)$, is clamped to a minimum of the target number of particles per cell, n. The same is done for cells on the domain boundary.

3.5 Variable Density

There are many examples of fluid flows involving fluids of variable density. For example, fluids undergoing thermal expansion will produce convection currents and concentration gradients of dissolved substances can result in variable density fluid flow, as can liquid-solid suspensions. In this section, we consider variable

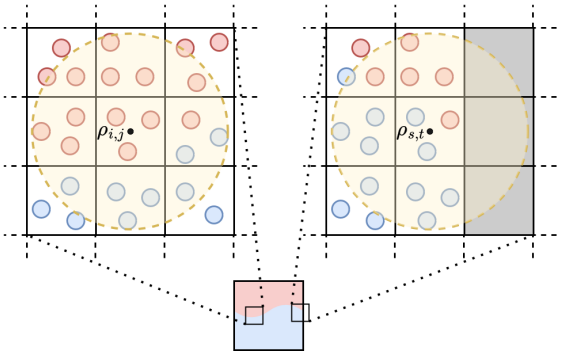

Fig. 2: D1-IDP and gIDP assume there is an even distribution of particles everywhere in the simulation domain. **Left**: $Cell_{i,j}$ lies within the body of the simulation domain so the density of the grid cell, $\rho_{i,j}$, is sampled from all particles throughout the entire interpolation kernel (yellow). **Right**: $Cell_{s,t}$ is at a solid boundary meaning some of the kernel contains no particles and so the density, $\rho_{s,t}$, is undersampled, leading to inaccuracies. A similar scenario occurs at the domain boundaries. In these cases, the density is clamped to a minimum value of the cell's target density.

density flows resulting from temperature differences. In the case of gIDP, this variable density will be considered in both the pressure PE and density PE, however, for D1-IDP this will only effect the pressure PE.

We use a simplified version of heat conduction as described by Stomakhin et al. [23]. Since we only deal with fluids, the simplifications avoid the changes of state between a liquid and solid. For this work, we do not alter the volume of a particle based on its temperature, only the density. In this sense, this aspect of the simulation breaks the conservation of mass.

Notation	Meaning
α	Fluid phase of parameter, where $\alpha \in \{A, B\}$
P_α	Number of particles beloging to phase α
K_p	Thermal conductvity of particle p
$K_{\alpha i}$	Phase α thermal conductivity of grid cell i
C_p	Heat capacity of particle p
$C_{\alpha i}$	Phase α heat capacity of grid cell i
T_p	Temperature of particle p
T_i	Temperature of grid cell i
m_p	Mass of particle p
$m_{\alpha i}$	Phase α mass of grid cell i
m_i	Total mass of grid cell i, $m_{Ai} + m_{Bi}$

Table 1: Notation relating to variable density fluids.

Alongside momentum, temperature is also interpolated between the particles and the grid. For heat conduction through the grid, we must interpolate the particle thermal conductivity and heat capacity to the grid as well as temperature (see Equations 9 - 11). Unlike velocity, grid temperature requires only a single scalar field for both phases, whereas thermal conductivity and heat capacity both require a separate field for each phase:

$$K_{\alpha i} = \frac{\sum_{p=0}^{P_\alpha} K_p m_p w_{ip}}{m_{\alpha i}} \tag{9}$$

$$C_i = \frac{\sum_{p=0}^{P} C_p m_p w_{ip}}{m_i} \tag{10}$$

$$T_i = \frac{\sum_{p=0}^{P} T_p m_p w_{ip}}{m_i} \tag{11}$$

where the notation used is described in Table 1. The cell temperature can then be updated as described by Stomakhin et al. in [23], making use of the phase face fractions in the differencing operator. Solid cells have a fixed temperature throughout the simulation and are labelled as either being a heat source or ambient.

Density is calculated using the standard formula for density change due to thermal expansion [24]:

$$\rho_\alpha = \rho_{\alpha 0}(1 - \kappa_\alpha \Delta T) \tag{12}$$

where $\rho_{\alpha 0}$ is the phase density at an ambient temperature, κ_α is the phase density coefficient, and ΔT is the change in temperature from the ambient state. Equation 12 is used for calculating the density of both particles and grid cells.

4 Results

To evaluate the volume preservation characteristics of gIDP and D1-IDP, we compare against MultiFLIP with both no additional volume preservation and the proportional controller method of [14]. The following three scenarios are presented:

- A bottleneck scenario, demonstrating a 'glugging' effect as water drains from a container through a narrow opening (see Figure 3). This comparison highlights the need for volume preservation in a multiphase PIC simulation.
- A Rayleigh-Taylor instability scenario, featuring a mixture of oil and water. As the liquids mix and separate a complex interface forms (see Figure 4). This will compare how effectively volume can be preserved in a simulation where the fluid interface has a large surface area compared to the volume of fluid.

– A variable density scenario, featuring liquids with varying densities based on temperature change (see Figure 5). This could be used to produce the effect that would be seen in a 'lava lamp'. This will examine the robustness of D1-IDP and gIDP, as different regions of the fluid interface will have different density ratios and each phases density will vary throughout the body of fluid.

For all simulations, unless stated otherwise, the parameters used are a timestep of $\Delta t = 0.001$, a grid cell size of $\Delta x = 0.5$, and a target number of particles per cell of $n = 8$ in the case of D1-IDP. For simulations using the proportional control method, we used a proportional gain of $k_p = \frac{1.15}{\Delta t}$ and an integral gain of $k_i = (\frac{k_p}{512})^2$, as given in [2]. In addition to this, we use FLIP transfers for each scenario. The simulations are run entirely on the CPU, and results have been produced using an Intel Core i9-13900K. All presented results are averaged over 3 simulation runs. The results have been rendered using Houdini. Additional parameter values for each of the presented scenarios is given in Table 2. A summary of the results for each scenario is presented in Table 3.

	Bottleneck	Rayleigh-Taylor	Variable Density
$\rho_A(g/cm^3)$	1.0	1.0	1.08
$\rho_B(g/cm^3)$	0.001	0.7	1.074
$\gamma(dyne/cm)$	72.0	35.0	150.0
$\kappa_A(K^{-1})$	-	-	0.0047
$\kappa_B(K^{-1})$	-	-	0.0028
$K_A(Wcm^{-1}K^{-1})$	-	-	0.146
$K_B(Wcm^{-1}K^{-1})$	-	-	0.596
$C_A(Jg^{-1}K^{-1})$	-	-	15.91
$C_B(Jg^{-1}K^{-1})$	-	-	39.93

Table 2: Parameter values used to generate the results demonstrated in this paper. A timestep of $\Delta t = 0.001$ was used for all simulations, and unless stated otherwise, the grid cell size used was $\Delta x = 0.5$. For simulations using D1-IDP, the target number of particles per cell was set to $n = 8$.

4.1 Bottleneck Scenario

This scenario showcases the interactions between water and air, with the water flowing through a narrow opening from the top container to the lower container of the same volume (see Figure 3). The density of the water is $1g/cm^3$, the density of the air is $0.001g/cm^3$, and $\gamma = 72.0dyne/cm$. The results demonstrate the glugging effect achieved when using a multiphase simulation in place of a single phase simulation. As can be seen in Figure 3, when no volume control measure is used the fluid suffers noticeably from a loss of volume over the course of the simulation, highlighting the need for some kind of volume preservation.

Scenario	Time per step (s)				Avg Vol Error (%)				Max Vol Error (%)				# Density Solver Iters	
	None	PC	gIDP	D1-IDP	None	PC	gIDP	D1-IDP	None	PC	gIDP	D1-IDP	gIDP	D1-IDP
Bottleneck	0.388	**0.384**	0.440	0.454	3.103	4.246	1.157	**0.022**	9.128	7.190	2.682	**0.144**	259.6	**145.2**
Rayleigh-Taylor	**0.363**	**0.363**	0.400	0.403	9.101	8.235	13.170	**0.255**	13.726	11.925	32.310	**0.597**	192.0	**152.4**
Variable Density	1.139	**1.135**	1.247	1.231	1.234	1.829	4.123	**0.421**	2.505	5.793	9.113	**0.659**	254.8	**220.0**

Table 3: A comparison of results for the 3 different presented scenarios. A timestep of $0.001s$, a grid cell size of $\Delta x = 0.5$ and a solver error threshold of $\epsilon = 10^{-16}$ were used for the given simulations. Volume error is calculated as $\frac{V(t)-V(0)}{V(0)}$. Only gIDP and D1-IDP have a density solver stage. Bold text indicates which method performed best for each metric.

When the proportional controller is used, the fluid initially loses volume similarly to when no volume preservation method is used. However, the integral gain term is able to recover some of the lost volume as the simulation progresses, resulting in a notable artifact of the fluid volume to 'growing'. Both gIDP and D1-IDP succeed in visually maintaining the volume throughout the duration of the simulation, and Table 3 shows us that the volume error is reduced. Table 3 also shows that D1-IDP is able to achieve a lower volume error when compared to gIDP, and additionally requires fewer iteration steps to reach the desired error threshold. The computational cost of using D1-IDP compared to gIDP is negligible, increasing the time per simulation step from $0.44s$ to $0.454s$.

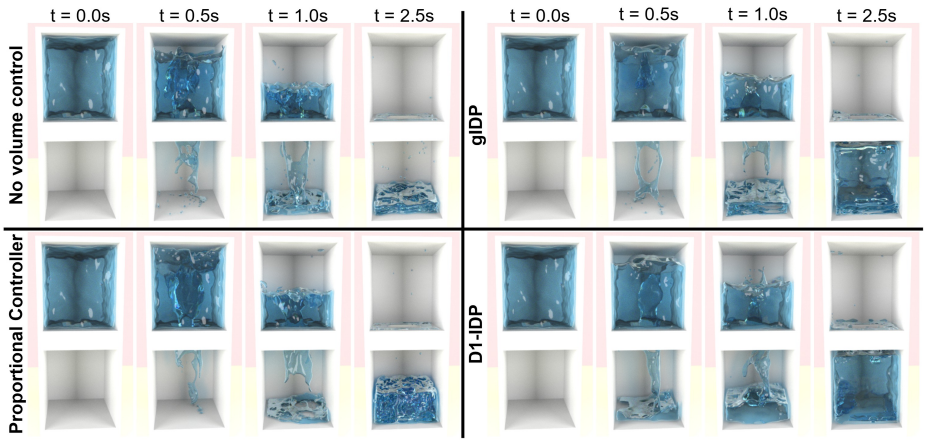

Fig. 3: A bottleneck scenario comparing volume preservation when using no volume control (top left), proportional control (bottom left), gIDP (top right) and D1-IDP (bottom right). Frames are taken from left to right at $t = 0.0s$, $0.5s$, $1.0s$ and $2.5s$.

We also tested the effect of the density solver error threshold on the volume preservation of gIDP and D1-IDP (see Table 4). As the error threshold decreases,

the number of solver iterations required to reach the threshold decreases for both gIDP and D1-IDP, as expected. As can be seen in the table, error threshold has a minimal effect on the volume preservation of both techniques, resulting in similar average and maximum volume errors for all tested thresholds. In the case of gIDP, as the error threshold is increased, the average volume error varies from a minimum of 0.447% for $\epsilon = 10^{-2}$ to a maximum value of 1.157% for $\epsilon = 10^{-16}$. Likewise, the maximum volume error varies from a minimum of 1.182% for $\epsilon = 10^{-2}$ to a maximum of 2.682% for $\epsilon = 10^{-16}$. A similarly small range is seen in the average and maximum volume errors when using D1-IDP, with a minimum average volume error of 0.006% for $\epsilon = 10^{-8}$ and maximum of 0.022% for $\epsilon = 10^{-16}$, and a range of 0.053% for $\epsilon = 10^{-8}$ to 0.211% for $\epsilon = 10^{-4}$ for the maximum volume error.

Furthermore, we also compare the effectiveness of gIDP and D1-IDP when using FLIP 0.9, APIC and AFLIP 0.9 (see Table 5). As shown, while using APIC reduces the efficacy of both gIDP and D1-IDP, the use of AFLIP improves the volume preservation of both methods.

We repeated these tests evaluating the effects of the density solver error and PIC method on both the Rayleigh-Taylor and variable density scenarios.

Scenario	Solver Error	Time per step (s)		Avg Vol Error (%)		Max Vol Error (%)		# Solver Iters	
		gIDP	D1-IDP	gIDP	D1-IDP	gIDP	D1-IDP	gIDP	D1-IDP
Bottleneck	10-16	0.440	0.454	1.157	**0.022**	2.682	**0.144**	259.6	**145.2**
	10-8	**0.433**	0.456	0.651	**0.006**	1.900	**0.053**	157.1	**85.6**
	10-4	**0.428**	0.451	0.812	**0.018**	1.626	**0.211**	95.3	**41.0**
	10-2	**0.425**	0.442	0.447	**0.015**	1.182	**0.095**	43.7	**20.1**
Rayleigh-Taylor	10-16	0.402	0.405	13.170	**0.256**	32.310	**0.597**	192.8	**153.0**
	10-8	0.403	**0.401**	5.726	**0.095**	15.343	**0.412**	123.9	**88.6**
	10-4	**0.398**	0.405	5.927	**0.080**	15.668	**0.336**	63.7	**52.5**
	10-2	**0.395**	0.396	4.461	**0.087**	11.469	**0.315**	37.6	**24.6**
Variable Density	10-16	1.247	1.231	4.123	**0.421**	9.113	**0.659**	254.8	**220.1**
	10-8	**1.244**	1.247	1.832	**0.029**	6.062	**0.124**	133.3	**123.5**
	10-4	**1.220**	1.231	2.838	**0.022**	5.917	**0.149**	68.9	**67.7**
	10-2	**1.204**	1.228	4.383	**0.030**	7.140	**0.145**	29.4	**29.2**

Table 4: A comparison of the effect of solver error threshold on the bottleneck scenario. The time step used was $0.001s$ and a grid cell size of $\Delta x = 0.5$. Volume error is calculated as $\frac{V(t)-V(0)}{V(0)}$. Only gIDP and D1-IDP have a density solver stage. Bold text indicates which method performed best for each metric.

4.2 Rayleigh-Taylor Instability Scenario

This scenario demonstrates interactions between oil and water, with a low density ratio and a smaller surface tension coefficient than the bottleneck scenario (see Figure 4). The density of the oil is $0.7g/cm^3$ and $\gamma = 35.0dyne/cm$, with the parameters for water being the same as in the previous section. As the fluids simulation progresses, a complex interface forms with a large surface area. As gIDP cannot be applied at the interface (see Section 3.2), its ability to resolve the

Scenario	Method	Time per step (s)				Avg Vol Error (%)				Max Vol Error (%)				# Density Solver Iters	
		None	PC	gIDP	D1-IDP	None	PC	gIDP	D1-IDP	None	PC	gIDP	D1-IDP	gIDP	D1-IDP
Bottleneck	FLIP 0.9	0.388	**0.384**	0.440	0.454	3.103	4.246	1.157	**0.022**	9.128	7.190	2.682	**0.144**	259.6	**145.2**
	APIC	**0.372**	0.383	0.434	0.437	16.974	16.693	7.461	**0.472**	9.129	7.150	17.997	**1.380**	240.3	**145.5**
	AFLIP 0.9	**0.374**	0.384	0.443	0.459	1.501	1.317	0.539	**0.011**	2.214	4.191	1.693	**0.074**	267.8	**145.1**
Rayleigh-Taylor	FLIP 0.9	**0.363**	**0.363**	0.400	0.403	9.101	8.235	13.170	**0.255**	13.726	11.925	32.310	**0.597**	192.0	**152.4**
	APIC	**0.351**	0.365	0.404	0.404	14.843	14.761	7.431	**1.65**	13.628	11.491	22.298	**2.702**	187.7	**151.7**
	AFLIP 0.9	**0.352**	0.361	0.41	0.411	5.375	7.623	6.234	**0.030**	10.677	10.124	14.507	**0.174**	219.6	**151.8**
Variable Density	FLIP 0.9	1.139	**1.135**	1.247	1.231	1.234	1.829	4.123	**0.421**	2.505	5.793	9.113	**0.659**	254.8	**220.0**
	APIC	**1.134**	1.137	1.220	1.237	2.992	3.058	10.677	**1.655**	2.502	5.795	25.085	**2.491**	271.7	**219.3**
	AFLIP 0.9	1.155	**1.145**	1.273	1.283	0.952	4.736	4.135	**0.028**	1.878	5.789	11.423	**0.145**	237.2	**221.1**

Table 5: A comparison of the effect of using different PIC methods for the bottleneck scenario. The time step used was $0.001s$, a grid cell size of $\Delta x = 0.5$ and a solver error threshold of $\epsilon = 10^{-16}$. Volume error is calculated as $\frac{V(t)-V(0)}{V(0)}$. Only gIDP and D1-IDP have a density solver stage. Bold text indicates which method performed best for each metric.

accumulating errors is limited, resulting in a maximum volume error of 32.60%. In contrast, D1-IDP is able to be applied across the interface, offering much better volume preservation in this scenario. However, D1-IDP outperforms the proportional controller approach, resulting in a 0.597% maximum volume error. As in the previous scenario, there is a negligible computation cost when using D1-IDP, this time resulting in an increase in the time per step of from $0.400s/step$ for gIDP to $0.403s/step$ for D1-IDP.

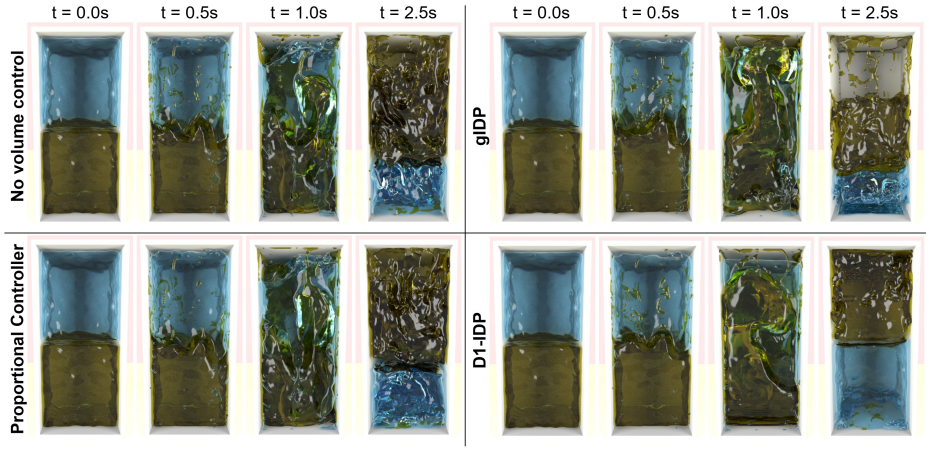

Fig. 4: A Rayleigh-Taylor scenario comparing volume preservation when using no volume control (top left), proportional control (bottom left), gIDP (top right) and D1-IDP (bottom right). Frames are taken from left to right at $t = 0.0s$, $0.5s$, $1.0s$ and $2.5s$.

4.3 Variable Density Scenario

This scenario shows a larger scene where both fluid phases have variable density based on the temperature (see Figure 5, Table 3). The floor of the container has a temperature of $200.0°C$, with the remaining walls fixed at $0.0°C$. Both fluids also begin the simulation with a temperature of $0.0°C$. As the simulation progresses, heat is conducted from the floor through both of the fluids, decreasing the density and producing a convection current.

The density of the orange liquid is $1.08g/cm^3$ at $0.0°C$ with a thermal density coefficient of $\kappa_A = 0.0047$, with the surrounding liquid being $1.074g/cm^3$ at $0.0°C$ with $\kappa_B = 0.0028$. The surface tension coefficient is set to $\gamma = 150.0dyne/cm$. These parameters are taken from the experimental data of Gyüre and Jánosi [9], but scaled to increase the speed of thermal conduction. These parameters result in the orange liquid heating up faster than the surrounding liquid, becoming less dense and floating to the surface. As it cools its density increases and it falls back down, generating a convection current. Similarly to the Rayleigh-Taylor scenario, a complex interface forms, affecting the performance of gIDP which achieves a maximum volume error of 9.113%. Again, since D1-IDP can be applied across the fluid interface the maximum volume error achieved is 0.659%. In this scenario, using D1-IDP results in a negligible decrease in the time per step required, decreasing from $1.247s/step$ for gIDP to $1.231s/step$ for D1-IDP.

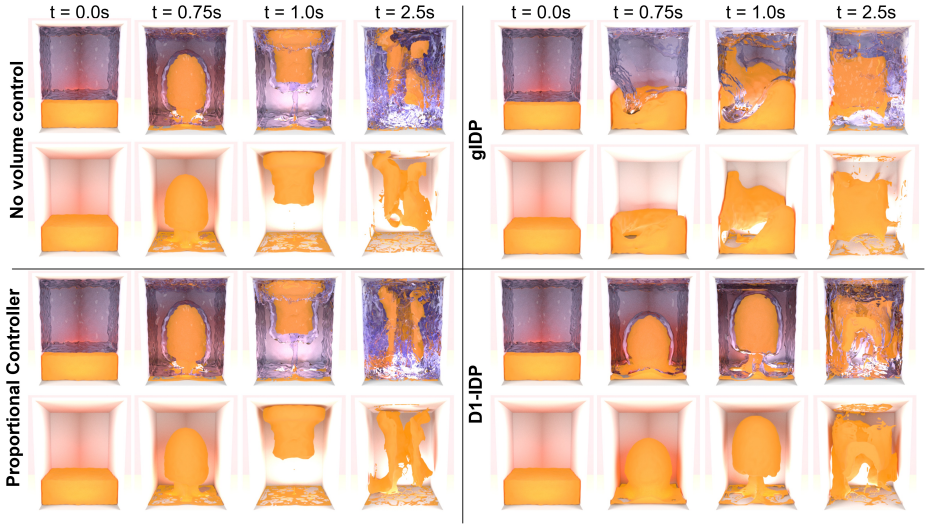

Fig. 5: A variable density scenario comparing volume preservation when using no volume control (top left), proportional control (bottom left), gIDP (top right) and D1-IDP (bottom right). In the bottom row of each section, the clear liquid has not been rendered so that the orange liquid can be seen more clearly. Frames are taken from left to right at $t = 0.0s$, $0.75s$, $1.0s$ and $2.5s$.

5 Discussion

As shown in the Section 4, the use of some kind of volume preservation for multiphase fluids is necessary to avoid severe volume loss. As can be seen in Tables 3, 4 and 5, D1-IDP improves upon the volume preservation of gIDP and the proportional controller in all of the given scenarios. This difference between gIDP and D1-IDP is especially noticeable in the Rayleigh-Taylor scenario – due to the fact that gIDP can't be applied at the fluid interface, the complex fluid interface that forms causes gIDP to perform poorly.

Both gIDP and D1-IDP have an increased computational cost when compared to using no volume preservation method or proportional control. When using the proportional controller, the bottleneck scenario takes $0.384s/step$ compared to $0.44s/step$ for gIDP (14.6% increase) and $0.454s/step$ for D1-IDP (18.2% increase) (see Table 3). Furthermore, D1-IDP generally results in a small increase in computational cost when compared to gIDP, with increases of 3.2% and 0.7% for the bottleneck and Rayleigh-Taylor scenarios, respectively. The largest increase in time per step of the given scenarios is 5.32% for the Bottleneck example with an error threshold of 10^{-8} and $\sim 76k$ particles.

D1-IDP also seems to smooth over finer details, such as small splashes, giving the appearance of a higher surface tension when compared to gIDP. This is most notable in the Rayleigh-Taylor demonstration, where the interface remains smoother throughout the duration of the simulation than when compared to using proportional control or gIDP (see Figure 4). The extent of this effect requires further investigation.

As stated in Boyd and Bridson [2], the positions of particles near the fluid interface are updated to prevent the phases from over-mixing. However, we do not update the positions of particles whose positions have already been updated during the particle bumping stage to avoid artifacts at the fluid interface. The interface bumping stage is likely a cause of volume loss as it does not consider the incompressibility of either fluid and can cause particle clumping (see Figure 1). Further work is required to investigate the interactions between these two position update stages.

6 Conclusions

We have presented Generalised Implicit Density Projection, an adaption of IDP that allows its application to multiphase fluid simulations using PIC methods. Additionally we have proposed D1 Implicit Density Projection, which improves upon the volume preservation characteristics of gIDP, especially in simulations with a complex fluid interface. We have shown that by treating both fluid phases as having a density of 1.0, D1-IDP can be applied across the fluid interface (Section 3.3), increasing the efficacy of the method. When comparing gIDP and D1-IDP, we have shown that across all given scenarios D1-IDP achieves a lower maximum volume error and results in fewer density PE solver iterations. This is true regardless of the solver error threshold used (see Table 4) and PIC method

(see Table 5). As previously mentioned, D1-IDP has a negligible impact on the computational cost when compared to gIDP, requiring a similar time per step for all of the presented scenarios.

As discussed in Section 5, updating the particle positions near the fluid interface is likely a cause of volume loss in multiphase fluids. Therefore, it would be interesting to investigate the impact of this bumping stage on volume preservation, as well as the interactions between the bumping and the particle position update in D1-IDP and gIDP. Additionally, it may be possible to combine the two position update stages into a single step which would potentially both reduce volume loss and preserve a smooth fluid interface.

Acknowledgments. This research is supported by a Frank Greaves Simpson Scholarship from The University of Sheffield.

References

1. Ando, R., Tsuruno, R.: A particle-based method for preserving fluid sheets. In: Proceedings of the 2011 ACM SIGGRAPH/Eurographics Symposium on Computer Animation. pp. 7–16. ACM (2011)
2. Boyd, L., Bridson, R.: MultiFLIP for energetic two-phase fluid simulation. ACM Transactions on Graphics **31**(2), 16:1–16:12 (2012)
3. Brackbill, J., Ruppel, H.: FLIP: A method for adaptively zoned, particle-in-cell calculations of fluid flows in two dimensions. Journal of Computation Physics **65**, 314–343 (1986)
4. Chen, S., Doolen, G.D.: Lattice boltzmann method for fluid flows. Annual Review of Fluid Mechanics **30**(1), 329–364 (1998)
5. Fei, Y.R., Guo, Q., Wu, R., Huang, L., Gao, M.: Revisiting integration in the material point method: a scheme for easier separation and less dissipation. ACM Transactions on Graphics **40**(4), 1–16 (2021)
6. Fu, C., Guo, Q., Gast, T., Jiang, C., Teran, J.: A polynomial particle-in-cell method. ACM Transactions on Graphics **36**(6), 1–12 (2017)
7. de Goes, F., Wallez, C., Huang, J., Pavlov, D., Desbrun, M.: Power particles: an incompressible fluid solver based on power diagrams. ACM Transactions on Graphics **34**(4), 1–11 (2015)
8. Guo, H., Wang, H., Zhao, J., Tang, Y.: Realistic rendering algorithm for bubble generation and motion in water. Electronics **11**(22), 3689 (2022)
9. Gyüre, B., Jánosi, I.M.: Basics of lava-lamp convection. Physical Review E **80**(4), 046307 (2009)
10. Harlow, F.H.: The particle-in-cell method for numerical solution of problems in fluid dynamics (LADC-5288) (1962)
11. Ihmsen, M., Cornelis, J., Solenthaler, B., Horvath, C., Teschner, M.: Implicit incompressible SPH. IEEE Transactions on Visualization and Computer Graphics **20**(3), 426–435 (2014)
12. Jeske, S.R., Westhofen, L., Löschner, F., Fernández-fernández, J.A., Bender, J.: Implicit surface tension for SPH fluid simulation. ACM Transactions on Graphics **43**(1), 13:1–13:14 (2023)
13. Jiang, C., Schroeder, C., Selle, A., Teran, J., Stomakhin, A.: The affine particle-in-cell method. ACM Transactions on Graphics **34**(4), 51:1–51:10 (2015)

14. Kim, B., Liu, Y., Llamas, I., Jiao, X., Rossignac, J.: Simulation of bubbles in foam with the volume control method. ACM Transactions on Graphics **26**(3), 98 (2007)
15. Koschier, D., Bender, J., Solenthaler, B., Teschner, M.: A survey on SPH methods in computer graphics. Computer Graphics Forum **41**(2), 737–760 (2022)
16. Kugelstadt, T., Longva, A., Thuerey, N., Bender, J.: Implicit density projection for volume conserving liquids. IEEE Transactions on Visualization and Computer Graphics **27**(4), 2385–2395 (2019)
17. Li, W., Liu, D., Desbrun, M., Huang, J., Liu, X.: Kinetic-based multiphase flow simulation. IEEE Transactions on Visualization and Computer Graphics **27**(7), 3318–3334 (2021)
18. Li, W., Wu, K., Desbrun, M.: Kinetic simulation of turbulent multifluid flows. ACM Trans. Graph. **43**(4), 55:1–55:17 (2024)
19. Lyu, L., Cao, W., Wu, E., Yang, Z.: Affine particle-in-cell method for two-phase liquid simulation. Virtual Reality & Intelligent Hardware **3**(2), 105–117 (2021)
20. Lyu, L., Ren, X., Cao, W., Zhu, J., Wu, E.: Adaptive narrow band MultiFLIP for efficient two-phase liquid simulation. Science China Information Sciences **61**(11), 114101 (2018)
21. Qu, Z., Li, M., De Goes, F., Jiang, C.: The power particle-in-cell method. ACM Transactions on Graphics **41**(4), 118:1–118:13 (2022)
22. Sancho, S., Tang, J., Batty, C., Azevedo, V.C.: The impulse particle-in-cell method. Computer Graphics Forum **43**(2), e15022 (2024)
23. Stomakhin, A., Schroeder, C., Jiang, C., Chai, L., Teran, J., Selle, A.: Augmented MPM for phase-change and varied materials. ACM Transactions on Graphics **33**(4) (2014)
24. Tipler, P.A.: Physics for scientists and engineers. Worth Publishers, New York, 3rd ed., extended version. edn. (1991)
25. Um, K., Baek, S., Han, J.: Advanced hybrid particle-grid method with sub-grid particle correction. Computer Graphics Forum **33**(7), 209–218 (2014)
26. Wang, X., Xu, Y., Liu, S., Ren, B., Kosinka, J., Telea, A.C., Wang, J., Song, C., Chang, J., Li, C., Zhang, J.J., Ban, X.: Physics-based fluid simulation in computer graphics: Survey, research trends, and challenges. Computational Visual Media **10**(5), 803–858 (2024)
27. Zhu, Y., Bridson, R.: Animating sand as a fluid. ACM Transactions on Graphics **24**(3), 965–972 (2005)