Implicit Density Projection for Volume Conserving Liquids

Tassilo Kugelstadt, Andreas Longva, Nils Thuerey, Jan Bender

Abstract—We propose a novel implicit density projection approach for hybrid Eulerian/Lagrangian methods like FLIP and APIC to enforce volume conservation of incompressible liquids. Our approach is able to robustly recover from highly degenerate configurations and incorporates volume-conserving boundary handling. A problem of the standard divergence-free pressure solver is that it only has a differential view on density changes. Numerical volume errors, which occur due to large time steps and the limited accuracy of pressure projections, are invisible to the solver and cannot be corrected. Moreover, these errors accumulate over time and can lead to drastic volume changes, especially in long-running simulations or interactive scenarios. Therefore, we introduce a novel method that enforces constant density throughout the fluid. The density itself is tracked via the particles of the hybrid Eulerian/Lagrangian simulation algorithm. To achieve constant density, we use the continuous mass conservation law to derive a pressure Poisson equation which also takes density deviations into account. It can be discretized with standard approaches and easily implemented into existing code by extending the regular pressure solver. Our method enables us to relax the strict time step and solver accuracy requirements of a regular solver, leading to significantly higher performance. Moreover, our approach is able to push fluid particles out of solid obstacles without losing volume and generates more uniform particle distributions, which makes frequent particle resampling unnecessary. We compare the proposed method to standard FLIP and APIC and to previous volume correction approaches in several simulations and demonstrate significant improvements in terms of incompressibility, visual realism and computational performance.

Index Terms—Fluid simulation, volume conservation, FLIP, APIC

1 Introduction

Over the last decades, fluid simulation has become an important tool in the visual effects industry. In recent years, it has also started to get relevant for interactive applications like games or virtual training simulations due to advances in algorithms and consumer hardware. Hybrid methods which combine Eulerian and Lagrangian approaches, like FLIP and MPM, are very popular as they combine the advantages of both viewpoints. They have been successfully used to simulate a large variety of materials like water, highly viscous fluids, sand, snow, deformable solids, and they perform especially well when it comes to interactions between different types of materials.

However, a central challenge for all of these methods is to enforce the incompressibility constraints of the underlying physical models. A major problem of the pressure projection, which enforces incompressibility, is that large time steps or large solver tolerances yield numerical volume errors that cannot be corrected by the pressure solver. These errors accumulate over time and lead to a visible loss of volume, resulting in visual artifacts that are obvious and disturbing for viewers. For long-term simulations and interactive scenarios these problems become particularly apparent. In practice, this forces users to accept tediously long runtimes induced by small time steps and large iteration counts.

In this work, we propose a novel method to track the fluid density by using the particles of the hybrid simulator. This enables us to measure the absolute compression or expansion of the fluid. We use the continuous mass conservation law to derive a pressure Poisson equation which enforces not only a divergence-free velocity field, but also constant density throughout the fluid. In this way our approach prevents volume loss and therefore improves the visual quality of the simulation results. Moreover, since our density projection method can correct errors that occur in the standard FLIP or APIC simulations, we can relax the strict solver accuracy and time step requirements. This speeds up simulations by a factor of up to 8 while producing visually comparable results without noticeable volume changes. Another benefit of the proposed method is that enforcing constant density also leads to more uniform particle distributions, which further improves the quality of the simulation results, and makes frequent resampling of the particles unnecessary. We also propose a way to robustly handle particles that accidentally enter solid obstacles — a common problem in FLIP and APIC simulations. It can be incorporated in the density projection method by applying Neumann boundary conditions so that the particle distribution is optimized globally and particles leave obstacles without being projected onto other fluid particles. Finally, our method can be easily incorporated into existing hybrid simulation methods like FLIP or APIC by extending the standard pressure solver.

In several comparisons of our approach with FLIP, APIC, and previous volume correction methods we demonstrate significant improvements in terms of incompressibility, visual realism and computational performance. Moreover, we show that our method can robustly handle large scale scenarios with complex boundaries by simulating scenes

- *Tassilo Kugelstadt - RWTH Aachen University*
  E-mail: kugelstadt@cs.rwth-aachen.de
- *Andreas Longva - RWTH Aachen University*
  E-mail: longva@cs.rwth-aachen.de
- *Nils Thuerey - Technical University of Munich*
  E-mail: nils.thuerey@tum.de
- *Jan Bender - RWTH Aachen University*
  E-mail: bender@cs.rwth-aachen.de

The definitive version of record is available at http://dx.doi.org/10.1109/TVCG.2019.2947437

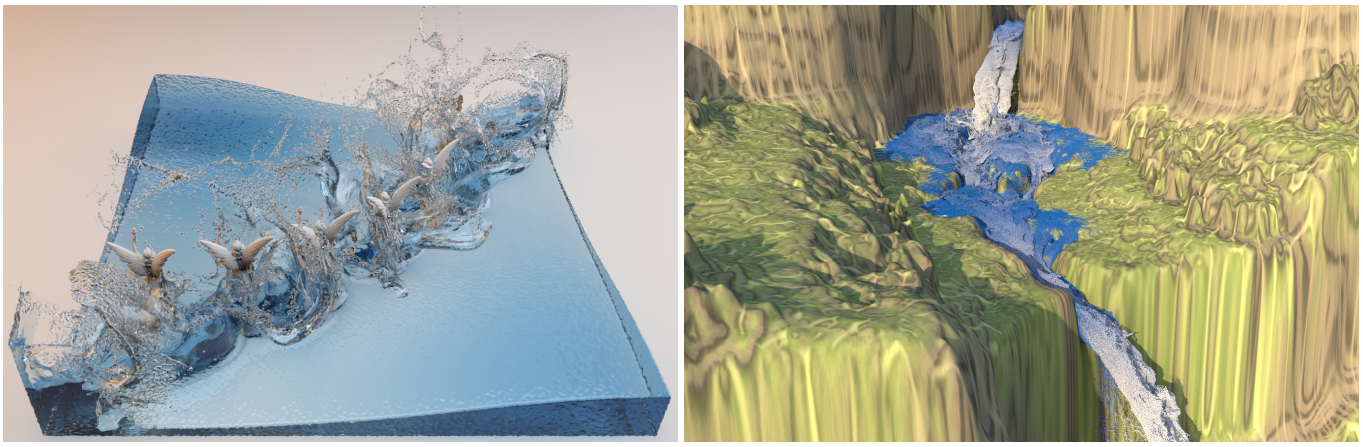


Fig. 1. Our implicit density projection method allows the efficient simulation of large-scale fluid scenarios while preventing undesired volume changes. Left: A double dam break with 8.8 million particles and 256^3 grid cells is hitting statues. Right: A complex river with up to 26 million particles and $1600 \times 400 \times 800$ grid cells is simulated at large time steps without noticeable volume loss.

with up to 26 million fluid particles. Finally, it enables us to produce realistic results without noticeable volume loss, even for large time steps (see Fig. 1) which improves the performance considerably. To summarize, our method yields significant quality and performance gains for a wide range of relevant liquid simulation scenarios, and is simple to integrate into existing solvers.

2 Related Work

Three-dimensional simulations of fluids were first employed in computer graphics by Foster and Metaxas [1]. Stam subsequently proposed the Eulerian *stable fluids* scheme [2], which was the basis for a popular class of liquid solvers with particle level sets and second order free surface boundary conditions [3], [4]. The *fluid implicit particle* (FLIP) method likewise combines grids and particles, and has been especially popular for detailed liquid simulations and visual effects productions [5]. The stable coupling of fluids with immersed bodies has been an important direction of work [6], [7]. More recently, generic approaches for coupling different solvers have also been proposed [8]. The pressure solve is a central part of Eulerian and hybrid solvers. It typically dominates their performance, and hence approaches such as dimensionality reduction [9], [10], fast iterative solvers [11] and efficient methods for grid-based adaptivity [12] have been proposed to reduce its runtime impact. For details regarding Eulerian fluid solvers we recommend the books by Bridson [13] or Kim [14].

The advection step of fluid simulations has received special attention, for example in the form of error correction schemes [15], [16] and schemes for conserving mass and momentum [17], [18]. The latter ones have also been used by Lentine et al. [19] to simulate liquids with very large time steps using the particle level set method. We instead focus on large time steps in hybrid Eulerian/Lagrangian simulations.

The hybrid algorithms originate from the *Particle in Cell* (PIC) method, in the context of which other researchers have proposed improvements in transferring quantities between particles and grid, such as the *Affine Particle in Cell* (APIC) [20] and *Polynomial Particle in Cell* [21] methods. Closely related, the *Material Point Method* (MPM), targets a wider range of material behaviors with a hybrid particle-grid approach similar to the FLIP method. While it was first proposed for snow simulations in the graphics context [22], it has since been extended to a wide range of material behaviors and simulation types [23], [24].

The FLIP algorithm itself has seen numerous extensions and improvements. For example, a narrow band particle placement was proposed to speed up calculations [25], [26]. In addition, researchers have noticed that the particle distribution of FLIP particles tends to cause problems over time. Several methods have aimed at alleviating this issue. Ando et al. [27] have proposed a position correction method inspired by SPH kernels. This method was extended by Um et al. [28] to sub-grid corrections. These approaches are closely related to our method because they can be used to prevent volume changes by correcting the particle positions. However, they are applying correction forces using explicit integration schemes which can cause an unstable behavior in a simulation with high stiffness values and large time steps as we show in our experiments. An implicit position correction has been proposed by Sato et al. [26], who use position based distance constraints to push particles apart when they are too close to each other. But their method is designed to correct the positions of particles in narrow band FLIP simulations and our experiments show that it cannot prevent volume loss in regular FLIP or APIC simulations with large time steps. Another approach for position correction was recently presented by Takahashi and Lin [29] which is based on position based fluids [30], an SPH method that enforces a constant density constraint. However, it requires costly particle neighborhood searches which can be avoided by our method where the particles only communicate indirectly via the grid.

The volume of fluids can be also controlled by adding divergence to the right-hand side of the Poisson equation in the pressure solve. This has been introduced by Feldmann et al. [31] to simulate the expansion of fluids in suspended particle explosions. Later it was used by Kim et al. [32] to control the volume of air bubbles. Their approach prevents a volume change of entire fluid regions, in their case each

air bubble. However, in the case of FLIP there are no distinct fluid regions, such as bubbles, so that this approach can be only applied globally. This is insufficient because the fluid usually gets compressed in some localized regions but the volume control will expand the whole fluid leading to worse particle distributions in formerly uncompressed regions. A local volume correction was proposed by Losasso et al. [33] in the context of a hybrid SPH and particle level set method. They use the mass conservation law to derive an additional term for the Poisson equation which penalizes too high particle numbers in local fluid regions. A similar approach has been proposed in the computational physics community by Liu et al. [34]. Gerszewski and Bargteil [35] applied this approach in FLIP simulations and refer to it as mass-full FLIP. In contrast to Losasso et al. they do not add an additional term to the right-hand side of the pressure Poisson equation because in the presence of strongly compressed fluids this can result in strong oscillations, popping and even explosions. To avoid these artifacts they use an additional solve on position-level and directly correct the particle positions without changing the velocity field as proposed by Narain et al. [36] for the simulation of granular materials or Irving et al. [37] for the simulation of incompressible deformable solids. Similar approaches are also popular in the SPH community [38], [39]. Our method is closely related to the one of Gerszewski and Bargteil who focused on the simulation of large scale splashing liquids. They use a unilateral incompressibility solve which is based on costly LCP solves to achieve realistic splashes. In contrast, our work focuses on the conservation of fluid volume which enables simulations with large time steps and lower solver accuracies and does not require expensive LCP solves.

3 Method

First, we briefly review the standard approach for fluid simulation and then discuss the volume conservation problem that we address in our work. An overview of our full simulation loop can be found as pseudo-code in Algorithm 1. The additional steps of our implicit density projection method are highlighted in blue and are derived and discussed in the following.

In the continuous theory, incompressible flows are modeled by the Navier-Stokes equation and the incompressibility condition

$$\frac{\partial \mathbf{u}}{\partial t} = -\mathbf{u}\cdot\nabla\mathbf{u} + \mathbf{g} + \nu\Delta\mathbf{u} - \frac{1}{\rho}\nabla p, \tag{1}$$

$$\nabla\cdot\mathbf{u} = 0, \tag{2}$$

where $\mathbf{u}$ denotes the velocity field, $\mathbf{g}$ the acceleration of gravity, ν the kinematic viscosity, ρ the density and p the pressure. Further, suitable boundary conditions are assumed as discussed in detail in the book of Bridson [13].

Most Eulerian methods assume that a divergence-free velocity field is sufficient to keep the fluid incompressible. But this only holds as long as $\nabla\cdot\mathbf{u} = 0$ is perfectly fulfilled at all times and at all positions inside the fluid. In practice, simulations have limited spatial and temporal resolution, and the accuracy of the pressure solver is also limited, and as a result this condition cannot be perfectly fulfilled. This can lead to density changes and undesired compression or expansion of the fluid. An even bigger problem is that the divergence of the velocity field only gives a differential view on density changes, but the absolute density error is invisible to the solver as we discuss below. This means that density errors which accumulate over time cannot be corrected by the pressure solver. Therefore, high accuracy of the linear system solver and sufficiently small time steps are mandatory to avoid volume errors. However, our experiments show that even when using small time step sizes, that are determined by a CFL number [13] smaller than 1, and solving the pressure system accurately, undesired volume errors in the fluid cannot be completely avoided. This leads to undesired visual artifacts. Moreover, the time step and accuracy restrictions are major reasons for the high computation costs of the simulation.

Algorithm 1 One step of FLIP with Implicit Density Projection. Additional steps for the density projection are highlighted.

1: Advect particles through grid
2: *Compute grid density using Eq.* (12)
3: *If necessary: handle degenerate configurations (see Sec. 3.2)*
4: *Apply boundary conditions as described in Sec. 3.3*
5: *Solve constant density PPE* (10) *with regular pressure solver*
6: *Compute position change* $\delta\mathbf{x}$ *using Eq.* (11)
7: *Correct particle positions by* $\delta\mathbf{x}$
8: Transfer velocity from particles to grid
9: Add velocity change due to body forces
10: Compute RHS of Eq. (9)
11: Apply boundary conditions
12: Solve divergence-free PPE (9)
13: Compute $\mathbf{u}(t+\Delta t)$ with Eq. (5)
14: Transfer velocities from grid to particles

3.1 Density Projection

While our main goal is to improve the visual quality of the simulation results by preventing undesired volume loss, we also want to overcome the strict time step and accuracy requirements in order to gain speedups. Therefore, we introduce a novel density projection method in this section. We track the density in the fluid by using the particles of the hybrid simulator. Density can be computed at each grid cell center by interpolating the particle mass onto the grid and dividing it by the cell volume, which is discussed in more detail in Section 3.2.

To see how this helps us on the theoretical side, we consider the mass conservation law:

$$0 = \frac{\partial \rho}{\partial t} + \nabla\cdot(\rho\mathbf{u}). \tag{3}$$

Here it becomes apparent that the divergence of the velocity field only measures density changes and not the absolute density of the fluid as we mentioned above. For incompressible fluids we need the additional constraint that the density of the fluid is constant

$$\rho = \rho_0 = \text{const}, \tag{4}$$

where ρ_0 is the rest density of the fluid. Usually this constraint is inserted into Eq. (3) so that the density derivatives

vanish, which results in the incompressibility condition $\nabla \cdot \mathbf{u} = 0$. In contrast, we continue with the principle of mass conservation to derive a pressure formulation that takes absolute density errors into account and is able to correct them. We discretize the Navier-Stokes equation (1) using the standard operator splitting approach

$$\mathbf{u}(t + \Delta t) = \mathbf{u}^* - \Delta t \frac{1}{\rho_0} \nabla p, \tag{5}$$

where $\mathbf{u}^*$ denotes the intermediate velocity field after applying advection and non-pressure forces. To compute the pressure, we discretize the time derivative in the mass conservation law (3) using backward Euler

$$\frac{\rho(t + \Delta t) - \rho^*(t)}{\Delta t} + \nabla \cdot [\rho(t + \Delta t)\, \mathbf{u}(t + \Delta t)] = 0, \tag{6}$$

where $\rho^*(t)$ denotes the intermediate density field. Plugging the velocity of the next time step from Eq. (5) and the incompressibility constraint $\rho(t + \Delta t) = \rho_0$ into Eq. (6) results in

$$\frac{\rho_0 - \rho^*(t)}{\Delta t} + \rho_0 \nabla \cdot \mathbf{u}^* - \rho_0 \Delta t \frac{1}{\rho_0} \nabla^2 p = 0, \tag{7}$$

which can be rearranged to

$$\frac{\Delta t}{\rho_0} \nabla^2 p = \nabla \cdot \mathbf{u}^* + \frac{1}{\Delta t} \left(1 - \frac{\rho^*(t)}{\rho_0} \right). \tag{8}$$

This yields a pressure Poisson equation (PPE) with an additional term on the right-hand side. Instead of only considering the divergence of the intermediate velocity field, we also take changes of the intermediate density field into account. This means that pressure not only counteracts compression that happens during one time step due to divergence, but also counteracts compression that already happened in the past. In the following subsections we will discuss spatial discretization, how to compute density and how boundary conditions are applied.

3.2 Discretization

The obvious way to discretize the PPE (8) would be to use standard MAC grids. However, our observations show that — especially in the presence of large density deviations — this can lead to strong oscillations, popping artifacts and even explosions. This has been reported by other researchers as well [33], [35], [36] but it becomes especially problematic in situations with large time steps and inaccurate pressure solves.

Splitting the PPE: In the context of granular materials Narain et al. [36] proposed to resolve the aforementioned issues with an additional solve to correct the densities on position level by instantaneously moving the particles without changing the velocity. Mathematically speaking the PPE gets split into two separate equations using the superposition principle [40]

$$\frac{\Delta t}{\rho_0} \nabla^2 p_1 = \nabla \cdot \mathbf{u}^*, \tag{9}$$

$$\frac{\Delta t}{\rho_0} \nabla^2 p_2 = \frac{1}{\Delta t} \left(1 - \frac{\rho^*(t)}{\rho_0} \right), \tag{10}$$

where Eq. (9) is the usual PPE that eliminates divergence from the intermediate velocity field. The second PPE (10) results in correction pressures p_2 that counteract compression and expansion. These two equations are discretized using standard MAC grids and can be solved with any pressure solver. Clearly, the sum of the exact solutions to the two equations solves Eq. (8) which can be easily seen by adding the equations and substituting $p_1 + p_2$ with p (this is also true when boundary conditions are applied cf. Sec. 3.3). However, now we have the possibility to treat the results of both PPEs differently. p_1 is used in the standard way to update the grid velocity field with Eq. (5) such that it becomes divergence-free. When we plug p_2 into Eq. (5), multiply by Δt and reorder the terms we get the position changes on the grid

$$\delta \mathbf{x} = \delta \mathbf{u} \Delta t = (\mathbf{u}(t + \Delta t) - \mathbf{u}^*)\, \Delta t = -\frac{\Delta t^2}{\rho_0} \nabla p_2. \tag{11}$$

To update the particles, $\delta \mathbf{x}$ gets interpolated at their positions and the particles are moved without changing the velocity.

This solves the problem of oscillations and popping artifacts. We believe the reason is that correcting the positions directly does not introduce additional divergence to the velocity field as solving Eq. (8) does. In the latter case the additional divergence is needed such that the density gets corrected during one time step. Afterwards the divergence has to be removed again by the pressure solver. Since this solve does not exactly remove the divergence completely due to numerical inaccuracies, this once again leads to density deviations and oscillations. Similar problems arise when solving hard constraints in rigid body simulations, where the strategy of solving constraints on a velocity level and applying a separate position correction has also been successfully applied [41], [42].

Details on the implementation of the pressure and density solve in the simulation loop can be found in Alg. 1. Since advection is the only step that changes the particle positions and therefore introduces density errors we correct the density directly after the advection step.

Density Computation: The density can be computed at the grid cell centers by mapping the particle mass m_p to the grid and dividing it by the cell volume V so that the density at grid cell i, j, k becomes

$$\rho_{i,j,k} = \frac{m_{i,j,k}}{V} = \frac{1}{V} \sum_p m_p N(\mathbf{x}_p). \tag{12}$$

Here, $m_{i,j,k}$ denotes the mass at the cell center which is interpolated from the particles at positions $\mathbf{x}_p$ with the interpolation kernel $N(\mathbf{x}_p)$. As a default, we use the common trilinear kernel. We also experimented with higher order kernels that are often used in MPM [43], but found that they lead to significant numerical dissipation. We initialize the particle mass so that we get rest density on the grid for uniform initial samplings. With N particles per cell the mass of each particle is set to $m_p = \rho_0 V / N$.

Particle Deficiency: The method described above gives accurate density estimates within the fluid body. But at the free surface and solid obstacles we have the problem that air and solid cells do not contain any particles, but they may be overlapped by the interpolation kernels from fluid cells.

This means that the density for the outermost fluid layer gets underestimated, which leads to undesired clumping of the particles that looks like artificial surface tension. In SPH this is known as the particle deficiency problem [44]. It can be overcome by clamping the density so that it cannot get smaller than the rest density. We only apply this clamping if at least one neighboring cell contains air. In the fluid, the particle attraction is actually desired, because there the density estimates are correct and undesired expansion of the fluid gets corrected. At solid walls, the particle deficiency can be overcome by also sampling the solids with particles. For non-moving objects, the mass has to be transferred to the grid only once which means that we get accurate density estimates there without any additional computation costs at runtime.

Limiting Displacements: We observed that large density errors can lead to very drastic correction displacements in a single time step, which can cause oscillations and popping artifacts. A simple way to avoid this is to clamp the right-hand side of Eq. (10). The intuition behind this is that the solver only sees a fraction of the density error which can be corrected safely in one time step. The remaining error will be corrected during the next time steps and oscillations are avoided. Our tests showed that clamping ρ^*/ρ_0 to the interval $[0.5, 1.5]$ limits the displacements so that the particles are not moved more than one cell width in one time step. Note that such extreme density deviations are rare in real-world applications.

Handling Degenerate Configurations: Our method can handle even degenerate particle configurations like very large numbers of particles in one grid cell (see Fig. 7). However, it is still possible that several particles are nearly at the same position so that they cannot be separated because they get the same interpolated position correction. To overcome this problem of conincidence we redistribute the particles in cells that contain far too many particles. As an indicator for the particle number per cell we use the density. When it is higher than $1.5\rho_0$ we redistribute the particles in the cell. For lower thresholds the redistribution is done more often and can impact the performance. For higher thresholds, the chance of missing coincident particles increases. The redistribution is done by splitting the cell into uniform subcells. Each particle is placed randomly in a region close to the center of a subcell to ensure the particles cover the whole volume of a cell. The new particle velocities are interpolated from the grid. As the redistribution only happens rarely and is limited to few cells, the computational overhead is negligible.

3.3 Boundary Conditions

The boundary conditions (BC) of the density projection are similar to the ones used in the pressure projection. At the fluid-air interface, we have the Dirichlet boundary condition $p_2 = 0$, meaning that the fluid can be moved into the air without any resistance. At fluid-solid interfaces one can use the Neumann BC $\delta\mathbf{x} \cdot \mathbf{n} = 0 \Rightarrow \nabla p_2 \cdot \mathbf{n} = 0$, where $\mathbf{n}$ is the normal vector of the solid surface. It prevents particles from being moved into solid obstacles and is identical to the BC for non-moving solids in the usual pressure solve.

Push-Out Boundary Condition: In addition to improving the conservation of volume, our formulation can be leveraged to enhance the treatment of solids in the flow. For solid walls the Neumann condition is typically used, and it works well as long as no fluid particles enter the solid obstacles. It is nonetheless a common problem in hybrid simulations that particles accidentally enter solid obstacles due to numerical errors in the advection and the grid-based velocity field. A standard approach is to project the particles back to the solid surface [5]. However, this can lead to particle clumping and volume loss, because the particles are moved into cells that might already have rest density or even too much mass inside them (see Fig. 9, top). Moreover, it is possible that several particles get projected to nearly the same position such that they cannot be separated because the interpolations from the grid result in the same velocities and displacements.

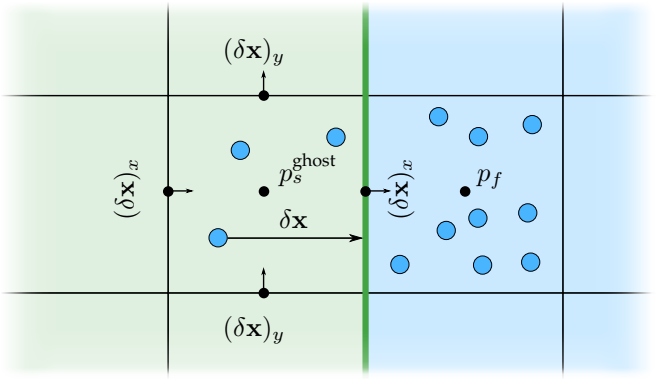


Fig. 2. *Push-out* boundary conditions applied to the velocity component between a solid cell s and a fluid cell f of a MAC grid. Blue cells denote fluid cells, green cells denote solid cells. The quantity $\delta\mathbf{x}$ is given by the distance of the particle with the deepest penetration ion the cell. The velocity boundary conditions at each face of the cell are set such that particles will tend to move out of the solid.

To overcome these problems we adapt the Neumann BC such that particles are pushed out of solid obstacles. In this way the density and therefore the particle distribution is optimized in a global fashion. When particles get moved out of the boundary into already filled cells, particles in these cells are also moved so that the density stays constant.

Usually solid objects are represented as signed distance fields (SDF). This means that the distance to the boundary $d(\mathbf{x})$ of a particle at position $\mathbf{x}$ can be easily determined by querying the SDF. Further, we can find the direction towards the closest point on the surface by computing the gradient of the distance function. Particles can be pushed out of the obstacle in one time step by the displacement

$$\delta\mathbf{x} = -\frac{\nabla d(\mathbf{x})}{\parallel \nabla d(\mathbf{x}) \parallel} d(\mathbf{x}), \tag{13}$$

where the negative sign comes from the convention that the distance value inside of objects is negative. If more than one particle is inside a solid cell, we use the $\delta\mathbf{x}$ of the one with the deepest penetration. The boundary displacement $\delta\mathbf{x}$ is set at all MAC faces of the cell (see Fig. 2). When particles are deeper than one cell inside the obstacle the displacements are set in the same way. There are no active pressure DOFs, so the pressure solve is not influenced. But when the displacements are applied to the particles afterwards, they get moved closer to the surface so that they get pushed out of the object in the following time steps. As

mentioned above, we want to avoid very drastic position corrections. Therefore, we clamp the displacement $\delta\mathbf{x}$ at the solid boundary to half of the cell width. This allows particles to leave solid objects on a stable path over several time steps, even for very deep penetrations (see Fig. 9, bottom).

The BC can be integrated into the solver by computing so called *ghost pressures* p^{ghost} for the solid obstacle cells [13]. They are denoted as ghost pressures since usually the pressure field is only defined for fluid and air cells but not for solid cells. The ghost pressure values can be determined by considering the pressure update Eq. (11) and discretizing the pressure gradient using finite differences. For one face of the MAC grid with index i, where one adjacent cell f contains fluid and the other cell s is solid, we get

$$\delta x_i = -\frac{\Delta t^2}{\rho_0}\frac{p_f - p_s^{\text{ghost}}}{\Delta x}, \tag{14}$$

where Δx is the grid cell spacing. This equation can be used to apply the BC so that the displacement of Eq. (13) is applied to push the particles out of the boundary. When we rearrange the equation, we can compute the ghost pressure for the solid cell as

$$p_s^{\text{ghost}} = p_f - \frac{\rho_0 \Delta x}{\Delta t^2}\delta x_i. \tag{15}$$

Next we consider the standard finite difference discretization of the Poisson equation (10) for the fluid cell f

$$\frac{\Delta t}{\Delta x^2}\left(\alpha p_f - p_s^{\text{ghost}} - p_n\right) = \frac{1}{\Delta t}\left(\rho_0 - \rho_f\right), \tag{16}$$

where $\alpha = 4$ in 2d and $\alpha = 6$ in 3d, and p_n contains the values of the remaining neighbors in the discrete 5 point (2d) or 7 point (3d) Laplacian stencil. Inserting p_s^{ghost} from Eq. (15) results in

$$\frac{\Delta t}{\Delta x^2}\left(\alpha p_f - p_f + \frac{\rho_0 \Delta x}{\Delta t^2}\delta x_i - p_n\right) = \frac{1}{\Delta t}\left(\rho_0 - \rho_f\right). \tag{17}$$

We can move the displacement term to the right-hand side yielding

$$\frac{\Delta t}{\Delta x^2}\left((\alpha - 1)p_f - p_n\right) = \frac{1}{\Delta t}\left(\rho_0 - \rho_f - \tfrac{\rho_0}{\Delta x}\delta x_i\right). \tag{18}$$

This shows that the push-out Neumann BC can be easily implemented by subtracting $\frac{\rho_0}{\Delta x \Delta t}\delta x_i$ on the right hand side. It is completely analogous to the Neumann BCs for moving solid obstacles in the pressure projection where we have to subtract the relative velocity. This means that we can use the standard pressure solver and only exchange the right hand side. The benefit of this boundary condition is evaluated in detail in Section 4.3.

4 Results

In this section we discuss results and compare our method to FLIP, APIC and previous volume correction methods in terms of visual quality and computational performance. Therefore, we implemented our method as a plug-in for the Mantaflow framework [45] which was used to create all presented simulations. It was straightforward to integrate our method into the main simulation loop, and in the same way it should also be possible to incorporate our method into any existing FLIP or APIC solver without much effort. In the following subsections and in the accompanying video, we present several simulation results to demonstrate the improvements of our method compared to FLIP and APIC. If not stated otherwise, we use a solver tolerance of $\varepsilon = 10^{-3}$ for the maximum norm of the residual, which is the default of Mantaflow. We start with a comparison of the volume conservation. Then we demonstrate that our approach leads to improved particle distributions and boundary handling. Finally, we present a performance comparison of FLIP and our method.

4.1 Volume Conservation

Incompressibility or conservation of volume is a very important visual feature of liquids like water. Therefore, we created several simulations to compare our method to FLIP and APIC in terms of incompressibility.

Volume Computation: In particle-based simulations volume is not exactly defined because the fluid is represented by a sparse set of points in space. Therefore, it is not possible to have an exact volume measure, and when we talk about volume, we mean that we computed it using the following approximation. In our initial sampling we have N_{init} particles in a completely filled fluid cell, which occupy the cell volume V. In the simulations we used $N_{\text{init}} = 3^2$ in 2d and $N_{\text{init}} = 2^3$ in 3d. This value is used to compute the portion of volume $V_{i,j,k}$ of the cell with index i, j, k that is actually covered by fluid as

$$V_{i,j,k} = \min\left(\frac{N_{i,j,k}}{N_{\text{init}}}V, V\right), \tag{19}$$

where $N_{i,j,k}$ denotes the number of fluid particles in the cell. This means that cells with N_{init} or more particles inside are completely filled and cells with less than N_{init} particles are only partially filled.

2d Double Dam Break: To evaluate volume conservation of FLIP and our method for different time step sizes Δt and pressure solver tolerances ε, we simulated a double dam break scene with a large variation of these solver parameters. In a first series of simulations we used a fixed solver tolerance of $\varepsilon = 10^{-3}$. The simulations were run with fixed time step sizes of $1\,\text{ms}$, $5\,\text{ms}$, $10\,\text{ms}$ and $20\,\text{ms}$. Here, $1\,\text{ms}$ is the smallest step size that a CFL number of 1 suggests during the whole scene.

The resulting simulations can be seen in the accompanying video, where deviation of the density from the rest density is color-coded from white indicating low deviation to red indicating high deviation. To evaluate the incompressibility quantitatively, we plotted the relative deviation of the total fluid volume from the initial volume in Fig. 3. As expected, our method conserves the fluid volume within the accuracy of the measurement, and the volume changes after 30 seconds of simulation are below 2% for all tested time step sizes. When using our method and a large time step size of $20\,\text{ms}$, the graph shows a volume loss during the first splash which is quickly corrected in the following frames. Note that the volume loss is not corrected immediately by our method since we clamped the right-hand side of Eq. (10) to improve the stability (see Section 3.2). In contrast, FLIP suffers from significant volume changes, which get worse when the time step size increases, and which accumulate

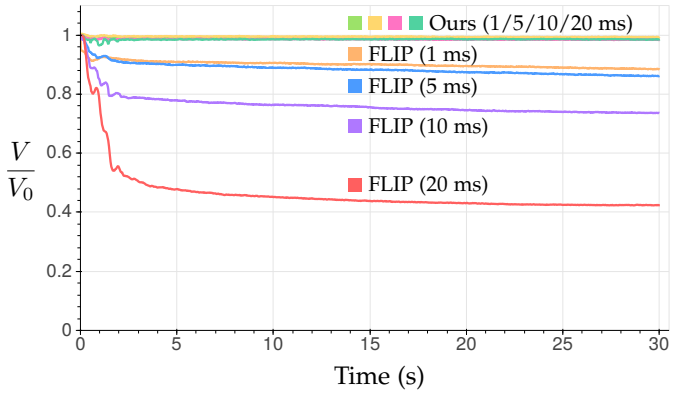


Fig. 3. Comparison of volume errors for FLIP and our method in a 2d double dam break with 71k particles, 128×128 grid cells, and a solver tolerance of 10^{-3}. The total fluid volume divided by the initial volume is plotted over time for several time step sizes. Our method keeps the volume change below 2% for all considered time steps. In contrast, FLIP suffers from significant volume loss, especially for large time steps.

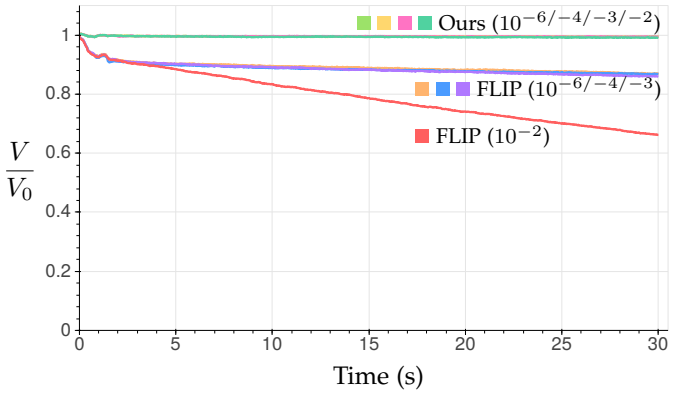


Fig. 4. Comparison of volume errors for FLIP and our method in a 2d double dam break with 71k particles, 128×128 grid cells, and a time step of $\Delta t = 5\,\mathrm{ms}$. The total fluid volume divided by the initial volume is plotted over time for several pressure solver tolerances. Our method keeps the volume change below 1% for all considered tolerances. In contrast, FLIP suffers from volume loss, especially for large tolerances.

over time. For $20\,\mathrm{ms}$ time steps, nearly 58% of the fluid volume gets lost during the 30 seconds of the simulation. For time steps of size $10\,\mathrm{ms}$ and $5\,\mathrm{ms}$ the fluid loses 26% and 14% of its volume, respectively. For these step sizes the volume deviation can be directly observed in the video, because it leads to lower water levels when the fluid comes to rest. In the simulation with $\Delta t = 1\,\mathrm{ms}$ the volume also deviates by 10% compared to the rest volume, but it leads to a rise in the water level. Here, the change comes from void regions inside the fluid body where FLIP produces a very uneven particle sampling (see Fig. 8). We will discuss the improvements that our method provides in more detail in Section 4.2. Further, we evaluate the dependence of the solver tolerance on incompressibility. Therefore, the 2d double dam break simulation is repeated with $\Delta t = 5\,\mathrm{ms}$ and solver tolerances of $\varepsilon = 10^{-2}$, $\varepsilon = 10^{-3}$, $\varepsilon = 10^{-4}$ and $\varepsilon = 10^{-6}$. The resulting simulations are shown in the accompanying video, and the relative volume deviation over time is plotted in Fig. 4. Our method is able to keep the volume nearly constant, and the volume change was less than 1% for all tested tolerances. In contrast to this, the FLIP simulation suffers from significant volume loss when the tolerance is too large. For $\varepsilon = 10^{-2}$ one third of the fluid volume gets lost during the simulation. For the stricter tolerance values the volume changes by roughly $13\% - 14\%$ and does not improve significantly for smaller ε values.

Comparison to Related Work: To compare our approach to previous volume correction methods we simulated the 2d double dam break with the parameters $\varepsilon = 10^{-3}$ and $\Delta t = 20\,\mathrm{ms}$ with various methods. One frame of the simulations is depicted in Fig. 5. First, we tried to solve Eq. (8) directly, which results in explosion-like artifacts as reported by Losasso et al. [33]. We implemented their solution which reduces the problem by averaging the density deviation on the RHS of Eq. (8) over a time interval of length τ. For small values of τ the artifacts are still present. Larger values reduce the artifacts but then the volume correction is not strong enough and the volume errors become visible. As proposed in [33] we use $\tau = 1\mathrm{s}$ for the scene in the video. It already suffers from significant volume loss and even for much larger values of τ there are artifacts.

We also implemented the method of Ando et al. [27] which uses SPH like weak spring forces to push the particles apart when they come too close to each other. This prevents volume loss but it suffers from stability issues due to the explicit integration. This results in undesired high frequency oscillations in the particle distribution. Decreasing the stiffness of the springs helps with this issue but then the volume errors become evident. We also compare to the correction method of Sato et al. [26] which uses implicit position based distance constraints to push the particles apart. However, this method was designed to improve the particle distributions near the surface in narrow band FLIP simulations and it is not capable of preventing the volume loss. In contrast to these previous methods, our approach successfully prevents volume errors without introducing artifacts. Moreover, we do not need particle neighborhood searches as Ando et al. and Sato et al.

3d Rotating Cuboid: To also evaluate the incompressibility in a more complex 3d scenario, we simulate a basin of water with a rotating cuboid in it. The scene contains 8 million particles and is simulated on a grid with 128^3 cells. The time step size is determined adaptively by using a CFL number of 1. The simulation was performed using FLIP and APIC, each with and without our method. The initial configuration and one frame after 30 seconds of the APIC simulation with and without our method is depicted in Fig. 6. In the FLIP / APIC simulation the fluid lost $38.5\%/38\%$ of its initial volume, which can be easily observed. With our method, the volume is preserved up to 0.5%, which is not visually noticeable.

Stability Tests: To test the stability of our method, we simulate two test scenarios. In the first one, we take a fluid basin in which the lower half is filled with 1 million particles and let the simulation run for a few seconds with a deactivated pressure solver. In this time, all fluid particles drop onto the floor. Then we reactivate the pressure solver. Without our method, nothing happens when the solver is reactivated, because the velocity field is already divergence-free and none of the lost volume is recovered. With our density projection, the entire fluid volume is stably recovered in less than a second. A side by side comparison of this

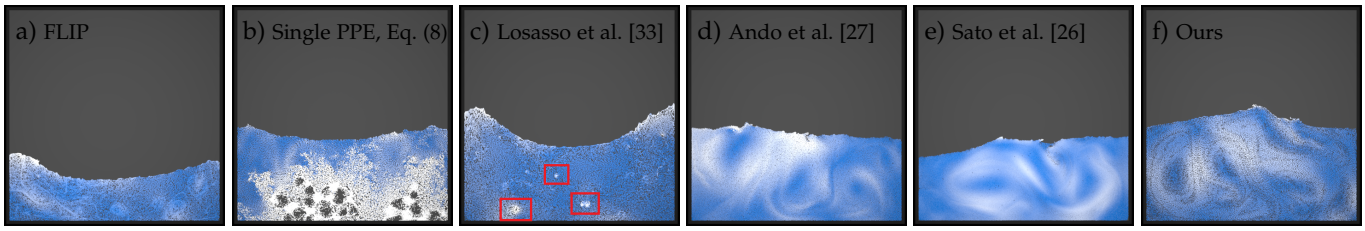


Fig. 5. Comparison to previous volume correction methods in a 2d double dam break scene. a) - f) show the same frame of simulations with different approaches. The velocities are color coded from blue (low) to white (high). a): Standard FLIP looses a significant portion of the fluid volume. b): Solving Eq. (8) results in explosion artifacts. c): The averaging approach of Losasso et al. [33] suffers from popping artifacts (cf. accompanying video). d): The weak spring position correction of Ando and Tsuruno [27] suffers from high frequency oscillations of the particles (cf. accompanying video). e): The distance constraints of Sato et al. [26] are not able to prevent the volume loss. f): Our method preserves the fluid volume and improves the particle distribution without artifacts.

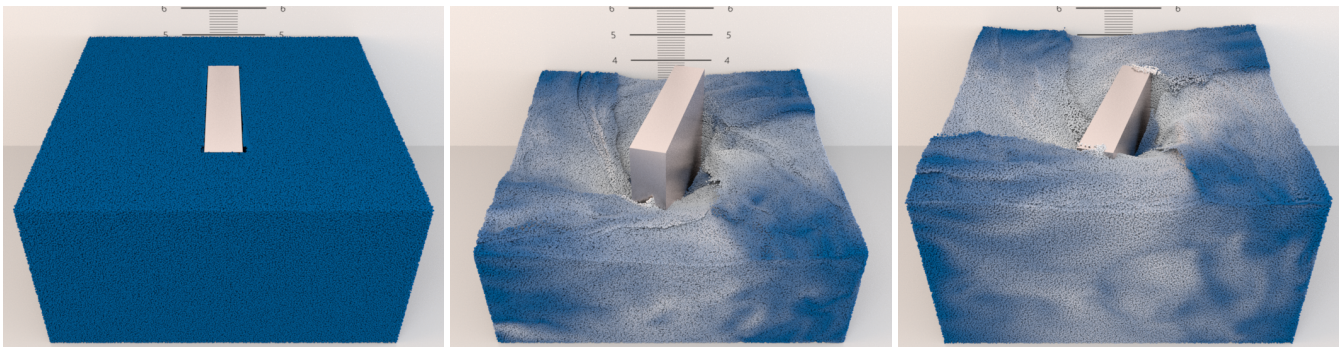


Fig. 6. Comparison of the volume conservation of APIC and our method in a complex 3d scene where fluid gets swirled around by a rotating cuboid. Left: Initial configuration. Middle: In the APIC simulation the water levels constantly drops during 30 second of simulation such that 38% of the fluid volume gets lost. Right: When simulated with APIC in combination with our density projection the volume change is below 0.5%.

simulation with and without our method can be found in the accompanying video.

In a second test we use the same scene, but instead of deactivating the pressure solver, we move the 1 million particles into a single grid cell as an initial condition for the simulation. Without our method, the FLIP simulation is not able to recover the fluid volume, and after a few seconds the fluid covers only a one grid cell thick layer on the floor. With our approach, the entire fluid volume stably recovers in less than one second. The initial condition and three frames from the recovering process are depicted in Fig. 7.

Complex Scenarios: We performed two simulations with a large number of particles and complex boundary geometry to verify that our method also works in practical scenarios. In the first one, we set up a double dam break with statues in the fluid basin in order to generate interesting splashes (see Fig. 1, left). The simulation runs on a 256^3 grid, contains 8.8 million particles and the time step is determined adaptively with a CFL number of 1. The second scenario is a river which is flowing through a complex canyon geometry with two waterfalls (see Fig. 1, right). This simulation is performed on a $100\text{m} \times 25\text{m} \times 50\text{m}$ domain which was discretized with $1600 \times 400 \times 800$ grid cells, and it contains up to 26 million particles. It runs stably with a large time step of $20\,\text{ms}$. These examples demonstrate that our approach can handle complex simulations efficiently and robustly without noticeable volume loss.

4.2 Particle Distribution

Another problem that is solved by enforcing constant density fields is the uneven particle sampling of regular FLIP and APIC simulations. This is demonstrated in Fig. 8, where we have taken one frame of the 2d double dam break simulation which we discussed in the previous subsection and zoom in to show the particle distribution. Here, density errors are color coded, where red refers to high errors, and white refers to low errors. The left part shows the regular FLIP simulation which contains particle clusters and void regions. They reduce the visual quality and lead to volume errors. The right part shows the simulation with our method. The particles are more regularly sampled which improves visual quality. The color coding shows that there are no observable density errors in the entire fluid volume.

4.3 Boundary Handling

In the following experiment we compare our push-out boundary conditions with the standard boundary handling of FLIP [5]. We simulate a basin of water in 2d which has a solid obstacle in the form of the Stanford bunny inside. As an initial condition, we sample not only the fluid but also the bunny with fluid particles. When we start the simulation using our approach, all particles leave the bunny in a fraction of a second in a stable way such that they contribute to the fluid volume and the water level rises. We also repeat this simulation without our method, in which case all particles are immediately projected to the surface of the bunny. However, then they clump together in a narrow band around the obstacle. This does not lead to a rise in

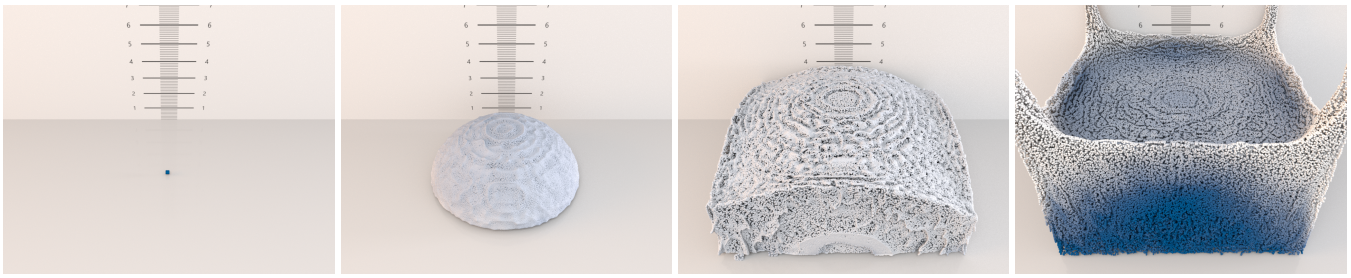


Fig. 7. Stability test for our method: We place 1 million particles in a single cell of a 64^3 grid and then start the simulation. With our method the fluid recovers its volume stably in less than a second. With regular FLIP the fluid cannot recover its volume.

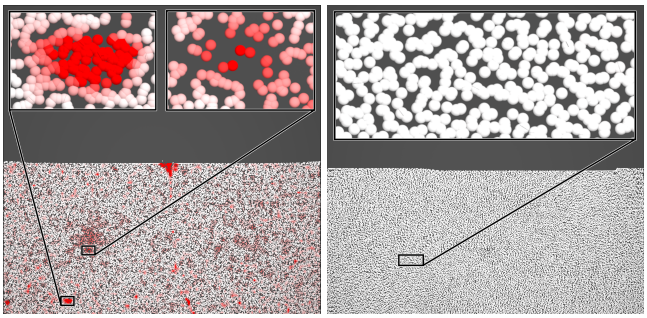


Fig. 8. Comparison of particle distributions of FLIP (left) and our method (right) in a 2d double dam break scene after the fluid has settled. The simulation contains 71k particles on a 128^2 grid. Density errors are color coded, red refers to high and white to low errors. Left: The regular FLIP simulation suffers from uneven particle distributions with particle clumping and void regions. This reduces the visual quality and results in unphysical volume changes. Right: Our method produces more regular particle distributions which increase the visual quality and avoid volume errors.

the water level, and the fluid volume inside the bunny gets lost. The initial condition and three frames of the simulation with and without our method are shown in Fig. 9. Thus, our method successfully recovers the liquid volume even for tough scenarios.

4.4 Computational Performance

We evaluate the computational costs of our method and compare them to standard FLIP by repeating a 3d double dam break simulation several times with different time steps and solver tolerances. It was performed on a $64 \times 128 \times 64$ grid and contained 275k particles. The simulation was run on a standard PC with an Intel Core i7 6700k quad-core CPU.

To evaluate the additional costs of our method, we run the simulation with the same parameters $\varepsilon = 10^{-4}$ and $\Delta t = 2\,\mathrm{ms}$ once with and without our method. Using standard FLIP, the simulation requires on average $188\,\mathrm{ms}$ per time step, while with our method it needs $242\,\mathrm{ms}$. This means that our density projection increases the computation cost per time step by 29%. However, we can actually achieve significant speedups by using larger time steps and a lower solver accuracy without losing visual quality and without drastic volume changes in the fluid.

Using our method, the simulation runs stably at $\Delta t = 20\,\mathrm{ms}$ and $\varepsilon = 10^{-3}$ with a volume change of 0.9%, which is visually not noticeable. When using the same parameters with FLIP, 35% of the fluid volume gets lost in 10 seconds. We repeated the FLIP simulation several times and decreased the time step and the solver tolerance until we did not see any improvements in the volume conservation. For $\Delta t = 2\,\mathrm{ms}$ and $\varepsilon = 10^{-4}$, FLIP still suffers from a volume change of 10% which did not improve with smaller time steps and tolerances. A side by side comparison of this simulation and the one using our method at $\Delta t = 20\,\mathrm{ms}$ and $\varepsilon = 10^{-3}$ is shown in Fig. 10. Using these settings, our method produces visually similar results while being 7.8x faster than the standard FLIP simulation. In general, scenarios can arise where using larger time steps is not an option because higher numerical damping can dissipate vorticity or details in the flow. However, in interactive or real time applications where the computation times are strictly limited, our method allows for significant speedups without volume loss. In future work we plan to further investigate the effect of large time steps on the kinetic energy and vorticity of the fluid and whether it can be improved, e.g. with fast energy projections [46] or micropolar models [47].

5 Conclusion

We presented an implicit density projection method which extends hybrid simulation methods, like FLIP or APIC, so that constant density can be efficiently enforced. This has the central advantage that volume errors that accumulate over time, and which are invisible to the regular pressure solver, can accurately and efficiently be corrected. In comparison to previous methods, our approach yields excellent conservation of volume without suffering from visual artifacts. This enables the use of larger times steps and less accurate pressure projections, which results in significant speedups. Further, it improves the particle distributions such that a frequent particle resampling is not required. Another benefit is that fluid particles can be pushed out of solid obstacles using our boundary conditions without introducing particle clumping and volume loss. In summary, our approach provides numerous benefits that lead to improved simulation quality and performance and is applicable to a wide range of practical scenarios.

Since our method is an extension of FLIP and APIC, it has the same limitations except for the ones that we addressed. One of these limitations is that it requires a dense particle sampling. Therefore, an interesting direction for future work would be a combination of our method with the narrow band FLIP approach of Ferstl et al. [25]

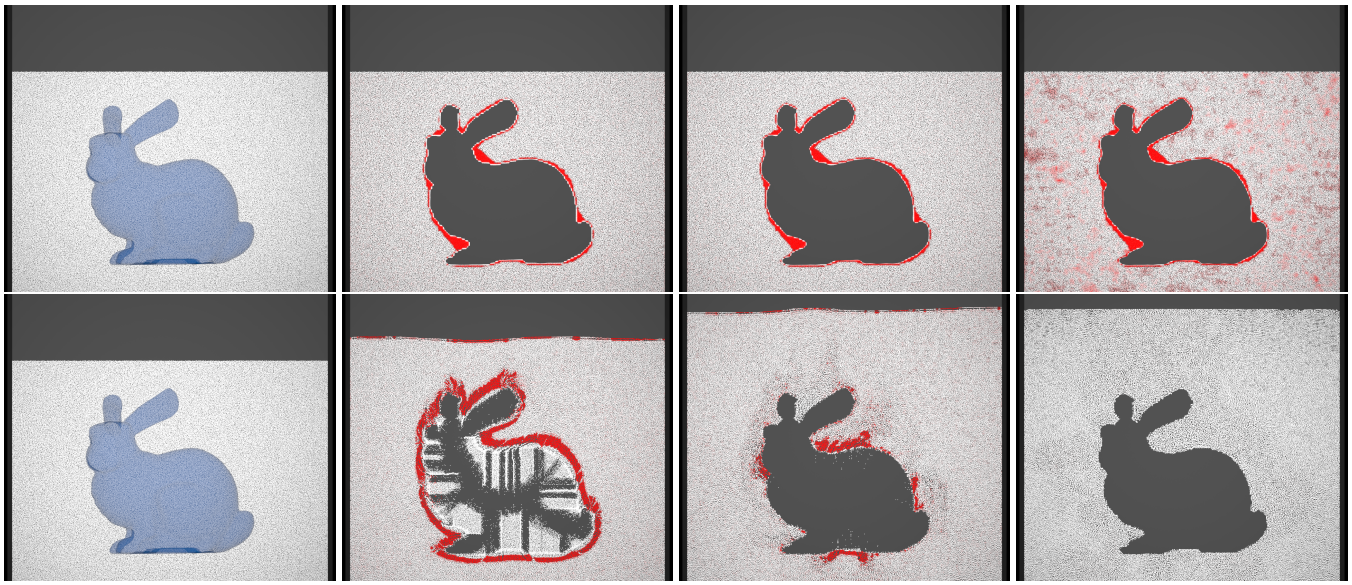


Fig. 9. Comparison of the standard FLIP boundary handling (top row) and our push-out boundary conditions (bottom row). The simulation is done on a 185^3 grid and contains 216k particles. Left column: As an initial condition we also sample a solid obstacle in form of a bunny with fluid particles. Top row: From left to right several frames of the regular FLIP simulation are depicted. The particles are projected to the surface where they are clumping together and the entire fluid volume that was inside the bunny gets lost. Bottom row: From left to right several frames of the simulation with our method are shown. The particle distribution gets optimized globally so that all particles leave the solid obstacle and the fluid volume is conserved. This can be observed as a rising water level.

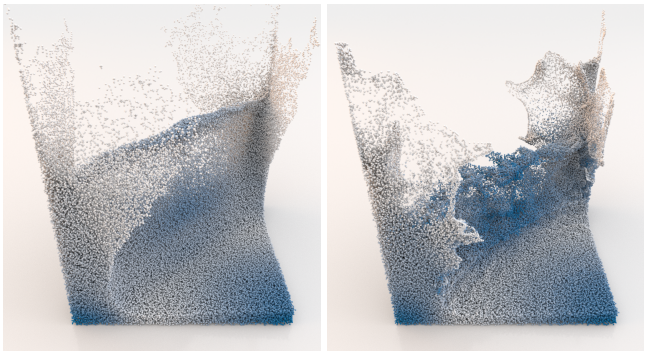


Fig. 10. Performance comparison of regular FLIP (left) and our method (right) in a double dam break with 275k particles and 64x128x64 grid cells. Our method robustly handles 10 times larger time steps and a higher solver tolerance than FLIP. It produces visually similar results without noticeable volume change while being 7.8 times faster.

to decrease the computation times. Moreover, we plan to investigate the applicability of our method in interactive applications by considering GPU implementations similar to the ones proposed by Wu et al. [48] or Gao et al. [49]. Finally, we have focused on liquid simulations in this work, but we plan to incorporate our approach into the material point method to simulate volume conserving deformable solids.

Acknowledgments

This work is supported by the German Research Foundation (DFG) under contract number BE 5132/4-1.

References

[1] N. Foster and D. Metaxas, "Realistic animation of liquids," *Graphical Models and Image Processing*, vol. 58, no. 5, 1996.

[2] J. Stam, "Stable Fluids," in *ACM Conference on Computer Graphics and Interactive Techniques*. ACM, 1999.

[3] N. Foster and R. Fedkiw, "Practical animation of liquids," in *ACM Conference on Computer Graphics and Interactive Techniques*, 2001.

[4] D. Enright, D. Nguyen, F. Gibou, and R. Fedkiw, "Using the Particle Level Set Method and a Second Order Accurate Pressure Boundary Condition for Free-Surface Flows," in *Proc. of the 4th ASME-JSME Joint Fluids Engineering Conference*, 2003.

[5] Y. Zhu and R. Bridson, "Animating Sand as a Fluid," *ACM Transactions on Graphics*, vol. 24, no. 3, 2005.

[6] C. Batty, F. Bertails, and R. Bridson, "A fast variational framework for accurate solid-fluid coupling," *ACM Transactions on Graphics*, vol. 26, no. 3, Jul. 2007.

[7] A. Robinson-Mosher, T. Shinar, J. Gretarsson, J. Su, and R. Fedkiw, "Two-way coupling of fluids to rigid and deformable solids and shells," *ACM Transactions on Graphics*, vol. 27(3), 2008.

[8] M. Akbay, N. Nobles, V. Zordan, and T. Shinar, "An extended partitioned method for conservative solid-fluid coupling," *ACM Transactions on Graphics*, vol. 37, no. 4, 2018.

[9] M. Lentine, W. Zheng, and R. Fedkiw, "A novel algorithm for incompressible flow using only a coarse grid projection," *ACM Transactions on Graphics*, vol. 29(4), 2010.

[10] R. Ando, N. Thuerey, and C. Wojtan, "A dimension-reduced pressure solver for liquid simulations," *Computer Graphics Forum*, vol. 34, no. 2, 2015.

[11] A. McAdams, E. Sifakis, and J. Teran, "A Parallel Multigrid Poisson Solver for Fluids Simulation on Large Grids," in *ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, 2010.

[12] M. Aanjaneya, M. Gao, H. Liu, C. Batty, and E. Sifakis, "Power diagrams and sparse paged grids for high resolution adaptive liquids," *ACM Transactions on Graphics*, vol. 36, no. 4, 2017.

[13] R. Bridson, *Fluid Simulation for Computer Graphics, Second Edition*. Taylor & Francis, 2015.

[14] D. Kim, *Fluid Engine Development*. AK Peters/CRC Press, 2017.

[15] A. Selle, R. Fedkiw, B. Kim, Y. Liu, and J. Rossignac, "An Unconditionally Stable MacCormack Method," *J. Sci. Comput.*, vol. 35, no. 2-3, Jun. 2008.

[16] J. Zehnder, R. Narain, and B. Thomaszewski, "An advection-reflection solver for detail-preserving fluid simulation," *ACM Transactions on Graphics*, vol. 37, no. 4, p. 85, 2018.

[17] M. Lentine, M. Aanjaneya, and R. Fedkiw, "Mass and Momentum Conservation for Fluid Simulation," in *ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, 2011.

[18] N. Chentanez and M. Müller, "Mass-Conserving Eulerian Liquid Simulation," in *ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. Eurographics Association, 2012.

[19] M. Lentine, M. Cong, S. Patkar, and R. Fedkiw, "Simulating free surface flow with very large time steps," in *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. Eurographics Association, 2012.

[20] C. Jiang, C. Schroeder, A. Selle, J. Teran, and A. Stomakhin, "The Affine Particle-In-Cell Method," *ACM Transactions on Graphics*, vol. 34, no. 4, Jul. 2015.

[21] C. Fu, Q. Guo, T. Gast, C. Jiang, and J. Teran, "A polynomial particle-in-cell method," *ACM Transactions on Graphics*, vol. 36, no. 6, 2017.

[22] A. Stomakhin, C. Schroeder, L. Chai, J. Teran, and A. Selle, "A Material Point Method for Snow Simulation," *ACM Transactions on Graphics*, vol. 32, no. 4, 2013.

[23] C. Jiang, T. Gast, and J. Teran, "Anisotropic elastoplasticity for cloth, knit and hair frictional contact," *ACM Transactions on Graphics*, vol. 36, no. 4, 2017.

[24] Y. Hu, Y. Fang, Z. Ge, Z. Qu, Y. Zhu, A. Pradhana, and C. Jiang, "A moving least squares material point method with displacement discontinuity and two-way rigid body coupling," *ACM Transactions on Graphics*, vol. 37, no. 4, 2018.

[25] F. Ferstl, R. Ando, C. Wojtan, R. Westermann, and N. Thuerey, "Narrow band FLIP for liquid simulations," *Computer Graphics Forum*, vol. 35, no. 2, 2016.

[26] T. Sato, C. Wojtan, N. Thuerey, T. Igarashi, and R. Ando, "Extended narrow band flip for liquid simulations," in *Computer Graphics Forum*, vol. 37, no. 2. Wiley Online Library, 2018, pp. 169–177.

[27] R. Ando and R. Tsuruno, "A particle-based method for preserving fluid sheets," in *ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. New York, NY, USA: ACM, 2011, pp. 7–16.

[28] K. Um, S. Baek, and J. Han, "Advanced hybrid particle-grid method with sub-grid particle correction," *Computer Graphics Forum*, vol. 33, no. 7, 2014.

[29] T. Takahashi and M. C. Lin, "A geometrically consistent viscous fluid solver with two-way fluid-solid coupling," in *Computer Graphics Forum*, vol. 38, no. 2, 2019, pp. 49–58.

[30] M. Macklin and M. Müller, "Position Based Fluids," *ACM Transactions on Graphics*, vol. 32, no. 4, 2013.

[31] B. E. Feldman, J. F. O'Brien, and O. Arikan, "Animating suspended particle explosions," in *ACM SIGGRAPH 2003*. ACM, 2003.

[32] B. Kim, Y. Liu, I. Llamas, X. Jiao, and J. Rossignac, "Simulation of bubbles in foam with the volume control method," in *ACM SIGGRAPH 2007*. ACM, 2007.

[33] F. Losasso, J. O. Talton, N. Kwatra, and R. Fedkiw, "Two-way coupled SPH and particle level set fluid simulation," *IEEE Transactions on Visualization and Computer Graphics*, vol. 14, pp. 797–804, 2008.

[34] J. Liu, S. Koshizuka, and Y. Oka, "A hybrid particle-mesh method for viscous, incompressible, multiphase flows," *Journal of Computational Physics*, vol. 202, no. 1, pp. 65–93, 2005.

[35] D. Gerszewski and A. W. Bargteil, "Physics-based animation of large-scale splashing liquids." *ACM Transactions on Graphics*, vol. 32, no. 6, pp. 185–1, 2013.

[36] R. Narain, A. Golas, and M. C. Lin, "Free-Flowing Granular Materials with Two-Way Solid Coupling," *ACM Transactions on Graphics*, vol. 29, no. 6, p. 1, 2010.

[37] G. Irving, C. Schroeder, and R. Fedkiw, "Volume Conserving Finite Element Simulations of Deformable Models," *ACM Transactions on Graphics*, vol. 26, no. 3, Jul. 2007.

[38] J. Bender and D. Koschier, "Divergence-Free SPH for Incompressible and Viscous Fluids," *IEEE Transactions on Visualization and Computer Graphics*, vol. 23, no. 3, 2017.

[39] S. Band, C. Gissler, M. Ihmsen, J. Cornelis, A. Peer, and M. Teschner, "Pressure boundaries for implicit incompressible sph," *ACM Transactions on Graphics*, vol. 37, no. 2, Feb. 2018.

[40] R. Haberman, *Applied Partial Differential Equations*, 4th ed. Prentice Hall, 2003.

[41] M. Cline and D. Pai, "Post-stabilization for rigid body simulation with contact and constraints," in *Proc. of IEEE International Conference Robotics and Automation*, vol. 3, 10 2003, pp. 3744 – 3751 vol.3.

[42] K. Erleben, "Rigid body contact problems using proximal operators," in *Proceedings of the ACM SIGGRAPH / Eurographics Symposium on Computer Animation*, 2017.

[43] M. Steffen, R. M. Kirby, and M. Berzins, "Analysis and reduction of quadrature errors in the material point method (MPM)," *International Journal for Numerical Methods in Engineering*, vol. 76, no. 6, 2008.

[44] M. Ihmsen, J. Orthmann, B. Solenthaler, A. Kolb, and M. Teschner, "SPH Fluids in Computer Graphics," in *Eurographics (State of the Art Reports)*. The Eurographics Association, 2014.

[45] N. Thuerey and T. Pfaff, "Mantaflow version 0.12," 2018. [Online]. Available: http://mantaflow.com/

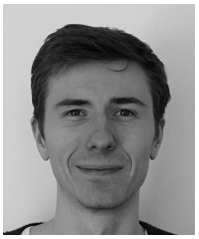
[46] D. Dinev, T. Liu, J. Li, B. Thomaszewski, and L. Kavan, "Fepr: Fast energy projection for real-time simulation of deformable objects," *ACM Transactions on Graphics*, vol. 37, no. 4, 2018.

[47] J. Bender, D. Koschier, T. Kugelstadt, and M. Weiler, "Turbulent micropolar sph fluids with foam," *IEEE Transactions on Visualization and Computer Graphics*, 2018.

[48] K. Wu, N. Truong, C. Yuksel, and R. Hoetzlein, "Fast fluid simulations with sparse volumes on the gpu," in *Computer Graphics Forum*, vol. 37, no. 2. Wiley Online Library, 2018.

[49] M. Gao*, X. Wang*, K. Wu*, A. Pradhana, E. Sifakis, C. Yuksel, and C. Jiang, "Gpu optimization of material point methods," *ACM Transactions on Graphics*, vol. 37, no. 6, 2018, (*Joint First Authors).

Tassilo Kugelstadt is a PhD student at RWTH Aachen University. He received his BSc degree in physics from JGU Mainz in 2013 and his MSc degree in Computer Science in the Natural Sciences from JGU Mainz in 2015. His research interests include physically-based simulation of deformable solids, elastic rods and fluids.



Andreas Longva is a PhD student at RWTH Aachen University. He received his MSc degree in Applied Mathematics from the Norwegian University of Science and Technology in 2017. His research interests encompass the physics-based simulation of deformable solids, rigid bodies and fluids, with a particular emphasis on finite element methods and numerical optimization.

Nils Thuerey is an Associate-Professor at the Technical University of Munich (TUM). He works in the field of computer graphics, where a central theme of his research are physics simulations and deep learning algorithms. He received a tech-Oscar from the AMPAS in 2013 for his research on controllable smoke effects. He worked for three years as a post-doc at ETH Zurich and as R&D lead at ScanlineVFX, before starting at TUM in October 2013.

Jan Bender is professor of computer science and leader of the Computer Animation Group at RWTH Aachen University. He received his diploma, PhD and habilitation in computer science from the University of Karlsruhe. His research interests include interactive simulation methods, multibody systems, deformable solids, fluid simulation, collision handling, cutting, fracture, GPGPU and real-time visualization.