Topology-Preserving Coupling of Compressible Fluids and Thin Deformables

JONATHAN PANUELOS, University of Toronto, Canada
EITAN GRINSPUN, University of Toronto, Canada
DAVID LEVIN, University of Toronto, Canada

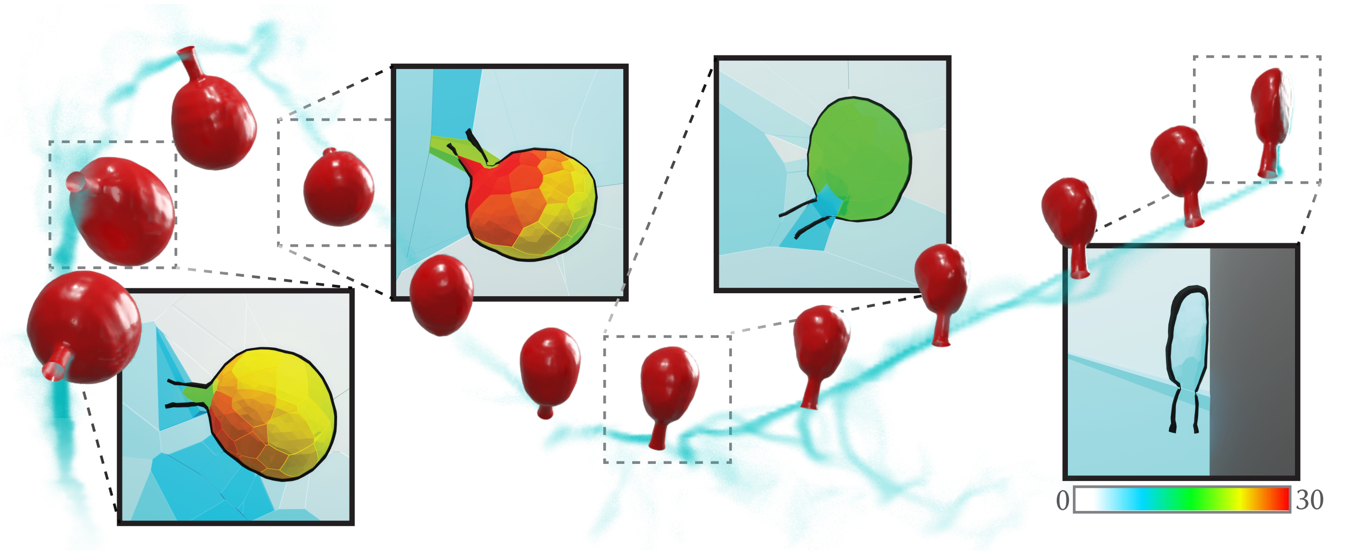


Fig. 1. A balloon being inflated then released to be propelled via the energy transfer between elastic potential energy to air pressure to kinetic energy. Insets show pressure value at each Voronoi cell inside and around the balloon.

We present a novel discretization of coupled compressible fluid and thin deformable structures that provides sufficient and necessary leakproofness by preserving the path connectedness of the fluid domain. Our method employs a constrained Voronoi-based spatial partitioning combined with Godunov-style finite-volume time integration. The fluid domain is discretized into cells that conform exactly to the fluid-solid interface, allowing boundary conditions to be sharply resolved exactly at the interface. This enables direct force exchange between the fluid and solid while ensuring that no fluid leaks through the solid, even when arbitrarily thin. We validate our approach on a series of challenging scenarios—including a balloon propelled by internal compressed air, a champagne cork ejecting after overcoming friction, and a supersonic asteroid—demonstrating bidirectional energy transfer between fluid and solid.

Additional Key Words and Phrases: Fluids, Liquids, Deformable Structures, Thin Shells, Voronoi, Compressible Euler Equations, Godunov, Finite Volume Method

1 Introduction

The interaction of compressible fluids with solids give rise to a range of varied and visually interesting phenomena. From a balloon whizzing through the air propelled by its own elasticity, to car suspension absorbing road shocks, or a stomp rocket launching skyward, the coupling of compressible fluids and solids holds great potential for compelling simulations hitherto overlooked in computer graphics. Beyond entertainment, these simulations have applications in areas such as modeling engine combustion chambers and capturing pressure gradients in biological systems like blood vessels.

While the general behaviour of fluids is governed by the Navier-Stokes equations, we focus our efforts specifically on the simulation of compressible inviscid fluids, which are given by the compressible Euler equations. Compared to incompressible fluids commonly simulated in computer graphics where pressure is often taken to be the Lagrange multiplier for the incompressibility constraint, compressible fluid treat pressure as an extra state variable in a trio of conservation laws for mass, momentum, and energy.

In the process of attempting to resolve the coupling of compressible fluids to solids, prior work would destroy the topology of the boundary between the two phases. They either insufficiently sample the boundary due to the difference in discretization between the solid and fluid phases, leading to leaking across thin boundaries, or sample these boundaries volumetrically thus adding thickness and potentially sealing flowable paths. We point out that in scenarios with highly dynamic solid structures, this is problematic as it could lead to considerably different and incorrect behaviour. Consider a self-propelled balloon for example, a leaky balloon might lose all its

Authors' Contact Information: Jonathan Panuelos, jonathan.panuelos@mail.utoronto.ca, University of Toronto, Toronto, Ontario, Canada; Eitan Grinspun, University of Toronto, Toronto, Ontario, Canada; David Levin, University of Toronto, Toronto, Ontario, Canada.

air before propelling itself, while an overly conservatively leakproof balloon may end up sealing its nozzle.

This motivates the need for a method that enforces *necessary and sufficient leakproofness*, preserving the *path connectivity* of the fluid domain. Fluid should be able to flow anywhere a continuous path exists through fluid, but never through solid. Our key insight here is that to preserve all potential flowable paths, thus allowing flow if and only if this connectivity exists, the solid and fluid discretizations must agree on where the solid is. That is, the solid geometry exactly must be a part of the fluid discretization.

We propose a novel fluid discretization that expands on prior Voronoi-based finite volume fluid discretizations [Springel 2010] by introducing the solid geometry as faces in the fluid discretization. Our new *volume stitching algorithm* ensures that the fluid mesh conforms precisely to solid boundaries, while preserving the fluid domain's topology. The result is an interface that is sufficiently and necessarily leakproof, ensuring that the discretized fluid domain respects the intended topology. Furthermore, the interface is sharply resolved at the solid boundary, enabling accurate force exchange between solid and fluid.

This partition can then be used for the explicit integration of the system, with fluid-fluid interactions being handled by the shared pairwise faces between any two particles, and the solid-fluid interactions are handled by any particles adjacent to the solid boundaries. Information is accurately passed between the two phases, from solid to fluid via a Dirichlet boundary condition on velocity, and from fluid to solid via a pressure force.

We demonstrate the versatility of our method across a range of challenging scenarios—including a balloon propelled by escaping air, a champagne cork ejecting under pressure, and a supersonic asteroid generating Mach cones—examples involving strong bidirectional coupling and complex topology. These examples showcase the method's ability to robustly handle thin structures, preserve fluid connectivity, and capture rich, visually compelling dynamics that were previously difficult to simulate within a unified framework.

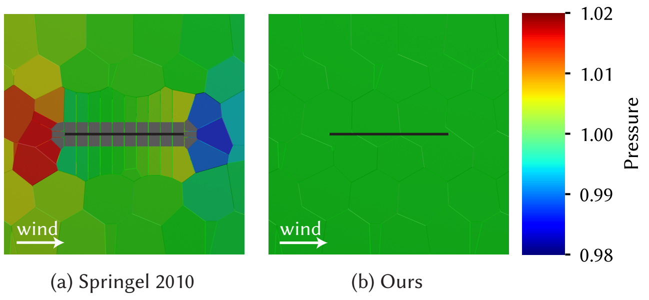


Fig. 2. Simulation of an infinitessimally thin sheet immersed in an inviscid fluid moving rightwards. Volumetric methods such as that of Springel [2010] produce nonphysical pressure variation at the leading and trailing edges of the sheet. Solid cells are shown in grey, and the solid surface is shown as a black line.

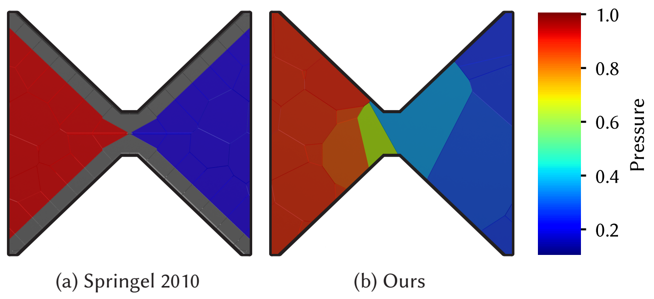


Fig. 3. Cutaway view of an hourglass-shaped narrow opening initialized with a Sod shock tube, with the high-pressure region on the left and low-pressure region on the right. Approaches using solid particles, such as that by Springel [2010], add additional thickness that closes the narrow opening, while our method resolves the solid *at* the specified surface. Solid cells are shown in grey, the boundary is shown as a black outline.

2 Related Work

2.1 Compressible Fluids

2.1.1 Lagrangian Methods. Much of the work in the simulation of compressible flow stem from the astrophysics community, which pioneered a Lagrangian description via Smoothed Particle Hydrodynamics (SPH) [Gingold and Monaghan 1977; Lucy 1977]. This involved a set of particles that track the fluid as it flows, with a kernel used for interpolation of physical quantities and their gradients onto the whole domain, allowing for low advection errors and automatic resolution adaptivity.

2.1.2 Eulerian Methods. In contrast, Eulerian methods partition space into *static* cells, with advection being handled via some interpolation scheme [Stone and Norman 1992]. In these methods, the system is often solved via finite-volume Godunov schemes, with numerical fluxes being computed at each interface between cells. Cells may be structured or unstructured, with the choice of domain being problem dependent. While fully gaseous applications, such as those intended by Stone and Norman [1992], typically focus on structured grids, applications that require accurate boundary handling, such as those in engineering, demand the use of unstructured meshes. Mavriplis and Venkatakrishnan [1997] presented a method for unstructured meshes consisting of both triangular and quadrilateral faces that conforms to input geometry (in their intended application, plane wings). Our discretization similarly respects input solid geometry, but in a Lagrangian setting where source points can be in arbitrary locations in the fluid domain. To achieve similar resolution adaptivity to Lagrangian methods, Berger and Colella [1989] introduced adaptive mesh refinement (AMR) for locally reducing grid size.

2.1.3 Moving Unstructured Meshes. Methods involving moving unstructured meshes attempt to achieve the advantages of both sides, with the high-accuracy flux and gradient computation of Eulerian schemes and accurate advection of Lagrangian. The domain is discretized using non-regular cells that move and deform according to the fluid motion. This grid motion is not necessarily required to follow the fluid velocity, allowing for an arbitrary Lagrangian-Eulerian (ALE) approach [Hirt et al. 1974], often used to avoid deteriorating mesh quality such as the presence of shards.

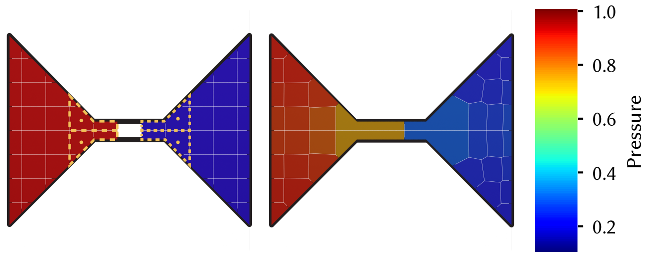
Börgers and Peskin [2005] introduced a Voronoi-based discretization, using it for incompressible flow, which was later adopted by Serrano et al. [2005] and Springel [2010] for compressible problems. The latter presented a volumetric approach to one-way solid coupling by explicitly including solid Voronoi points, but such volumetric approaches fail to properly resolve thin solids, as shown on Figure 2, as well as potentially sealing small openings, shown on Figure 3. Our work adopts their fluid discretization in the bulk flow, and presents a method for handling bidirectional solid coupling by modifying the Voronoi partition in the regions around the solid boundaries.

In contrast with the Voronoi representation of fluids, Voronoi partitions have also been used to represent solids in cell lattice methods, with Hwang et al. [2021] presenting a coupling scheme with weakly-compressible SPH fluids. We once again emphasize the limitations of such a volumetric representation of solids.

2.1.4 Moving Meshless Methods. With similar objectives, moving meshless methods have been developed, where source points partition space, but via fields of overlapping weights rather than strict mesh allocations [Lanson and Vila 2008]. These differ from SPH in that they define a partition of unity over all space and still apply Godunov-like interface fluxes [Hopkins 2015].

2.1.5 Riemann Solvers. As part of Godunov-type finite volume schemes, at each interface, the computation of a stable numerical flux is required. The first, and most diffusive, approximation is due to Lax [1954], which is equivalent to taking an average over an entire cell's domain. Kurganov and Tadmor [2000] reduces this domain of dependence to within the maximum signal velocity of waves propagating from the interface, reducing diffusivity. Various other approximate solver have been proposed, such as the Roe linearization [Roe 1981], HLLE [Einfeldt 1988; Harten et al. 1983], HLLC [Toro et al. 1994], as well as exact solvers relying on Newton iteration [Toro 2013], but we note that our method is agnostic to the choice of Riemann solver.

2.1.6 In Computer Graphics. Much of the effort in fluid simulation in computer graphics has largely been focused on incompressible flow. SPH in particular, although having its roots in the highly compressible regime, has been adapted by the community for incompressible flow, via the application of different particle potentials [Desbrun 1996] or via a pressure solve [Bender and Koschier 2016]. Some efforts have been done to soften the divergence constraint and allowing for controlled deviations from a defined rest density [Becker and Teschner 2007; He et al. 2025], but this work is limited to the weakly compressible regime. The closest in this field to our work is that of Cao et al. [2022], which simulate true compressible fluids in the context of supporting fluid shock-induced solid fracturing. Due to their choice of an MPM discretization, their solids are inherently volumetric and as such are limited by the resolution of the MPM grid. Our method, in comparison, conforms the fluid discretization around the solid, and are able to resolve even codimensional solids.



(a) Grétarsson 2013 (b) Ours

Fig. 4. 2D Sod shock tube through a narrow opening, with the boundary shown in black. The mass lumping of Grétarsson and Fedkiw [2013] around the tube is shown in dotted yellow, discarding cells not adjacent to cell centers (yellow circles).

We achieve this surface-respecting discretization via a modification of the Voronoi diagram induced by the Lagrangian fluid particles. The Voronoi diagram has been prior leveraged in computer graphics for representing fluid domains, albeit not compressible ones. Incompressible flow was simulated by Sin et al. [2009], computing the divergence free condition locally on each Voronoi cell. Brochu et al. [2010] added an explicit surface tracker that allows for computing more accurate surface forces such as surface tension. Saye and Sethian [2011] similarly leveraged it for surface tracking for multiphase flows, aiding in simplifying topology changes. Unlike these works, we consider every interface between each fluid particle, with physics being computed directly on the Voronoi diagram, rather than on a background Eulerian mesh. Voronoi diagrams have also been used to simulate foam bubbles, where each bubble is a radius-restricted Voronoi cell, with inter-bubble interfaces being represented by faces of the Voronoi diagram [Busaryev et al. 2012; Qu et al. 2023].

2.2 Solid-Fluid Coupling

The domain of fluid-structure interaction (FSI) in mechanics literature is rich in methods for coupling fluid simulation to solids. Most notable among these are arbitrary Lagrangian-Eulerian (ALE) interfaces [Donea et al. 1982], the ghost fluid method [Fedkiw et al. 1999], and the immersed boundary (IB) method [Peskin 1972].

2.2.1 Arbitrary Lagrangian-Eulerian. methods discretize the fluid domain via a deformable mesh that attempts to match the mesh representation of the solid while retaining a fully static Euler representation far away from the solid [Donea et al. 1982]. The key difficulty in these methods is computing a valid deformation of the mesh without degrading mesh quality, thus limiting the method to small solid deflections of volumetric solids. Recent advancements have been introduced to support incompressible fluids with codimensional solids with larger deformations via auxiliary coarse meshes, but deformations are still constrained and the method is limited to 2D and simpler geometries [Fernandes et al. 2019]. These methods are thus not suitable to the significant deformations we present in our examples.

2.2.2 *Ghost Fluid Method.*

Fedkiw et al. [1999] introduced the ghost fluid method to provide sharp interface handling for Eulerian multiphase fluid simulations. Ghost values were introduced into the numerical stencil, modifying accessed data to enforce boundary conditions across interfaces, usually tracked via level sets. The method was later extended from multiphase fluids into interfaces with Lagrangian solids [Fedkiw 2002]. This work also became the preferred method for solid boundary handling in computer graphics literature [Bridson 2015].

A similar method was adopted for boundary handling in SPH by filling solid domains with fictitious particles to avoid kernel deficiencies [Randles and Libersky 1996; Takeda et al. 1994], which was also later adopted by the graphics community [Schechter and Bridson 2012]. While this method is similar in spirit to the original ghost fluid, it loses sharp interface tracking due to the inherent smoothness introduced by SPH, as well as being necessarily volumetric.

More recent work has introduced its use in arbitrary polyhedral finite volume discretizations [Vukčević et al. 2017]. The key difficulty in these discretizations, shared from the original ghost fluid method, is the mismatch between the solid boundary and the fluid parcel boundary. We point out that the boundary condition becomes considerably simpler in our case because we match the fluid boundary to the solid boundary. We can simply reflect fluid particles across the face normal to enforce zero flux penetration across the solid faces.

2.2.3 *Immersed Boundary Method.*

embeds Lagrangian solid boundaries within a background Eulerian fluid grid, transferring forces and velocities by smoothing out the Lagrangian representation via delta functions [Peskin 1972, 2002]. Although initially developed for simulation of codimensional heart valves [Peskin 1972], no guarantees were made in terms of leakproofness. In particular, because only the solid is treated as Lagrangian points, with the fluid being discretized on a static grid, the solid often passes by fluid control points. In an incompressible fluid, as was the original implementation, this is not an issue as pressure and density remain constants throughout, but this offers fluid sidedness tracking challenges in compressible flows. Additionally, the original formulation which blurs solid velocity across a finite thickness does not enforce exact velocity matching with the fluid, and can thus violate no-flux conditions.

Extensions were developed to allow for sharp enforcement of boundary conditions, both in incompressible [Mittal et al. 2008] and compressible [Ghias et al. 2007] flow. These largely involve cut-cell and ghost cell approaches, effectively slicing the Eulerian grid to restrict it to one side of the domain. We highlight the cut-cell methodology presented by Ye et al. [1999] (once again for incompressible flow), involving extending boundary grid cells to include empty cells missing their grid points cut off by solid boundaries. These effectively modify the grid, warping the square grid into trapezoids wherever they are cut off by the solid boundary. Grétarsson and Fedkiw [2013] used a similar approach for compressible flow, and are able to handle thin solids, but recognized that their method may discard volumes that cannot be attached to adjoining grid cells, as shown on Figure 4. Our stitching algorithm is conceptually similar to these approaches, reattaching orphaned fluid parcels to an existing fluid source point, and can be considered as a generalization of their method into unstructured meshes. We demonstrate that our approach overcomes their limitations, and are able to resolve even subgrid fluid paths.

2.3 Modified Voronoi Diagrams

Visibility-constrained Voronoi diagrams have been extensively studied in computational geometry as generalizations of classical Voronoi tessellations to environments with occlusion or directional visibility restrictions. For instance, Aurenhammer et al. [2014] analyze diagrams where each site is restricted to a visibility wedge, resulting in Voronoi cells that may be non-convex, disconnected, and of quadratic combinatorial complexity. A broader treatment is provided in the monograph by Okabe et al. [2000], which surveys visibility-aware and obstacle-constrained variants in applications ranging from robotics to spatial statistics.

However, these constructions typically resolve endpoints of obstacles as Voronoi source points. In contrast, our setting involves codimensional solid boundaries that *lack* any Voronoi sources on their surface. As a result, traditional visibility-constrained approaches cannot be directly applied. Using such methods adds thickness on solid vertices, potentially destroying the topology of the fluid domain. Instead, we modify the Voronoi diagram induced solely by fluid particles by explicitly clipping it to the solid geometry and then reassigning orphaned cells via a path-connectivity-preserving stitching algorithm. This constructively embeds solid interfaces into the fluid partition, enabling sharp boundary resolution and leakproof coupling—features not addressed by existing visibility-constrained methods.

Tsin and Wang [1996] does present a method for a Voronoi diagram where barriers only constrain visibility and do not induce their own sites, but their work is limited to rectilinear barriers in 2D. The generalization to arbitrary barriers, as well as a further generalization to 3D, is nontrivial.

3 Equations of Motion

We aim to solve the fluid flow as described by the compressible Euler equations, which are a hyperbolic conservation law

$$\frac{\partial}{\partial t}\mathbf{U}(\mathbf{x},t) + \nabla \cdot \mathbf{F}(\mathbf{x},t) = \mathbf{0}\,, \tag{1}$$

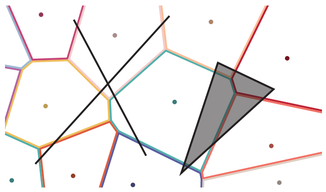
where $\mathbf{U}$ represents the vector of conserved quantities

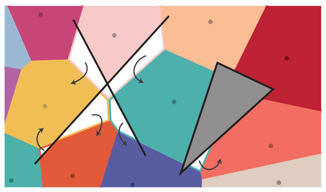
$$\mathbf{U} = \begin{bmatrix}\rho & \rho u_x & \rho u_y & \rho u_z & \rho e_T\end{bmatrix}^\mathsf{T}\,, \tag{2}$$

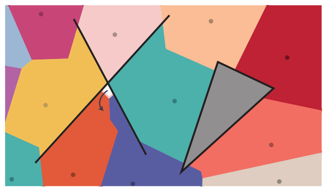
consisting of mass density ρ, momentum density $\begin{bmatrix}\rho u_x & \rho u_y & \rho y_z\end{bmatrix}^\mathsf{T}$, and total energy density ρe_T, and $\mathbf{F}$ is a nonlinear convective flux

$$\mathbf{F} = \begin{bmatrix}\rho u_n \\ \rho u_x u_n + P n_x \\ \rho u_y u_n + P n_y \\ \rho u_z u_n + P n_z \\ \rho u_n\left(e_T + \frac{P}{\rho}\right)\end{bmatrix} \tag{3}$$

governing the flow of these conserved variables across some unit normal vector $\mathbf{n} = \begin{bmatrix}n_x & n_y & n_z\end{bmatrix}^\mathsf{T}$ arising from the velocity $\mathbf{u} = \begin{bmatrix}u_x & u_y & u_z\end{bmatrix}^\mathsf{T}$ and normal velocity $u_n = \mathbf{u} \cdot \mathbf{n}$. The specific total energy $e_T = e + \frac{1}{2}||\mathbf{u}||^2$ is the sum of internal energy e and kinetic energy $\frac{1}{2}||\mathbf{u}||^2$.


(a) Initial Voronoi diagram (coloured outlines), clipped by solid constraints (black outlines, black shaded).


(b) Cells containing a source point are assigned (filled), while the rest are registered as "orphaned" (white). Orphan cells will inherit the valid neighbours with the largest face area (grey arrow).


(c) New connectivity reveals more orphaned cells that can be inherited.

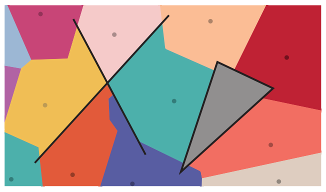

(d) Final modified Voronoi diagram.

Fig. 5. Stitching orphaned cells back to valid cells from the clipped Voronoi based on path-connectivity constraints.

Observe the explicit presence of pressure P, rather than the usual Lagrange multiplier treatment used in incompressible simulation.

This system is underdetermined and requires an equation of state to complete it. We use the ideal gas law $P = (\gamma - 1)\rho e$, but note that the method presented is agnostic of the chosen state equation. We take the adiabatic index γ to be 1.4, representing ideal diatomic gases.

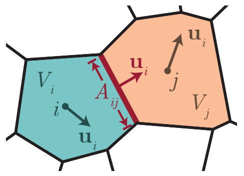
Via Green's theorem, we can rewrite the conservation law as

$$\frac{\partial}{\partial t}\int_{\Omega} \mathbf{U}(\mathbf{x},t)\, dV + \int_{\partial\Omega} \mathbf{F}(\mathbf{x},t)\cdot \mathbf{n}\, ds = \mathbf{0}, \tag{4}$$

which holds for any choice of fluid parcel Ω in space. This weak form guides our choice of discretization and naturally motivates the use of finite volume partitions, where numerical fluxes are computed at cell interfaces to enforce conservation locally.

4 Topology-Preserving Discretization

We seek a discretization that resolves boundary conditions imposed by arbitrarily shaped solids, including thin shells and other codimensional interfaces. This rules out fixed-resolution methods and volumetric solid representations, which require excessive refinement to capture thin structures. We also require support for large solid deformations—such as those of a deflating balloon—without compromising the fluid discretization, excluding static meshes and deformable mesh methods with fixed connectivity (e.g., ALE), which are typically limited to low-deformation volumetric solids.



These constraints naturally point to Lagrangian fluid representations. Among these, we prioritize compatibility with finite volume methods to sharply resolve solid interfaces. We adopt the Voronoi-based discretizations of Börgers and Peskin [2005] and Springel [2010], in which Lagrangian particles track the fluid motion and induce a spatial partition, assigning each particle the region closest to it in space.

Applying this discretization to the weak form in Equation 4 gives the fluid state update for some particle i with nearest-neighbour particles j,

$$\frac{\partial}{\partial t}\mathbf{U}_i + \sum_j \frac{A_{ij}}{V_i}\hat{\mathbf{F}}_{ij}\cdot \mathbf{n}_{ij} = \mathbf{0}, \tag{5}$$

where $\mathbf{U}_i$ is the fluid state of particle i, $\hat{\mathbf{F}}_{ij}$ is a numerical flux computed between i and j, A_{ij} is the area of their interface, n_{ij} is the outwards pointing normal of said interface, and V_i is the volume associated with i. The geometry of this interface between particles i and j is shown in the inset above.

This finite-volume equation where a numerical flux is evaluated at each interface is known as the Godunov form.

4.1 Challenges to Leakproofing

Significant care needs to be taken near boundaries, given that in our Godunov-type discretization (Equation 5), forces are applied only along faces of the Voronoi partition. In order to properly couple any boundary condition with the fluid particles, we must modify the Voronoi diagram to include the boundaries as edges in the diagram. Additionally, given a particle on one side of the barrier, its given domain should not extend past a barrier unless there is a contiguous path through the fluid to that side. In other words, a particle's domain must represent a single contiguous fluid.

Springel [2010] applied the solid boundaries as explicit particles that induce their own Voronoi cells. This has the disadvantage of being a volumetric representation of the solid, and thus cannot accurately capture arbitrarily thin structures.

Figure 2 demonstrates that the "thickening" approach is poorly suited to for immersed thin structures. We simulate a flat sheet immersed in a tangentially flowing fluid. Because the inviscid fluid flow is always parallel to the thin solid surface, the solid should not induce any change in the fluid state. Unfortunately, the volumetric approach requires a finite thickness inducing a pressure build-up on the leading edge of the sheet and a low-pressure wake on the trailing edge.

Furthermore, as depicted in Figure 3, the "thickening" approach is poorly sited for flow through small orifices, as it is prone to numerical sealing of small holes, such as those found in funnels, medicine droppers, jet sprays, or balloon nozzles. Our method resolves orifices, no matter how small, without introducing additional particles to represent the solid surface.

A similar issue arises from the work of Grétarsson and Fedkiw [2013] for a different reason. Their discretization utilizes a static grid, and performs mass lumping to reallocate partial cells to neighbouring grid degrees of freedom. This has the issue of potentially discarding partial cells, as shown on Figure 4.

Algorithm 1 Clipped Voronoi Stitching

```
 1: Compute initial Voronoi tessellation
 2: Clip Voronoi cells by solid boundaries
 3: for each cell do
 4:     if cell does not contain its generating point then
 5:         Label cell as orphaned
 6:     end if
 7: end for
 8: while any cell is orphaned do
 9:     for each orphaned cell do
10:         for each neighboring non-orphaned cell do
11:             Compute interface area between cells
12:         end for
13:         Merge orphaned cell with neighboring non-orphaned cell with largest interface area
14:         Update orphaned cell label
15:     end for
16: end while
```

Therefore, we seek a discretization that exactly recovers the connectivity produced by the solid boundaries, is leakproof where and only where required, and allows for fluid flow wherever a valid path exists.

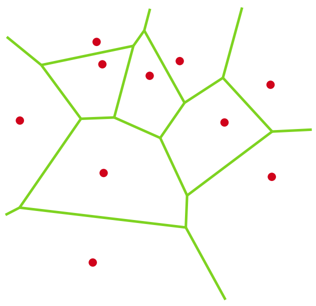
4.2 Clipped Voronoi Stitching Algorithm

We develop a stitching algorithm that augments the set of interfaces from the Voronoi diagram with the faces of the solid boundary, reassigning patches of space to the appropriate fluid particle in a path-connected manner, so that fluid may continue to flow wherever a valid path exists around the solid.

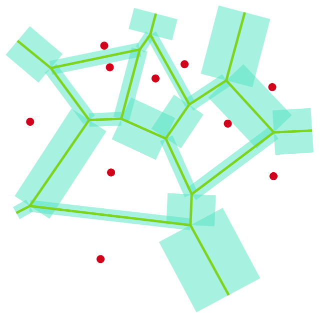
Our method is initialized by constructing the standard Voronoi diagram induced by the fluid particles over all of space. We then include all solid faces into this structure by clipping all the cells with these faces. Any Voronoi faces inside volumetric solids will be removed, while codimensional solid faces will be added as new faces in the diagram. Each fluid face is then tagged with the label of the bordering fluid cells, constructing a neighbourhood graph where each edge is a valid fluid-only path through space.

Around these solid faces, a number of cells will become "orphaned", *i.e.* they will no longer contain a source point within the cell. We tag these cells, then for each orphaned cell, we will assign it to its non-orphaned neighbour with the largest interface area, as shown on Figure 5. We choose to attach by largest interface area due to our choice of finite-volume as the integration method. Since fluxes are computed pairwise at each interface weighted by interface area, orphaned cells ought to be most influenced by whichever neighbour it shares the largest face with. This procedure is done iteratively, as some orphaned cells may be landlocked by other orphaned cells until said cells are assigned.

The completed algorithm is presented on Algorithm 1. Our orphan-cell reassignment is conceptually similar to breadth-first region growing or flood fill, but adapted to prioritize interface area and operate on clipped Voronoi cells, preserving finite-volume structure and fluid path connectivity under solid constraints.



(a) Voronoi mesh



(b) Flux domains of dependence at each interface

Fig. 6. A Voronoi diagram at some timestep (left), and the domains of dependence of the waves propagating at each interface (right). Voronoi mesh is represented in green, with source points shown in red. Domains of dependence are shown in light blue.

4.3 Necessary and Sufficient Leakproofing

It is straightforward to show that our method preserves the connectivity of the fluid domain as constrained by the input solid boundaries.

First, solid boundaries are included as extra faces on top of the original Voronoi diagram. Second, a fluid face is never deleted unless it is fully inside a volumetric solid.

Third, a fluid face split by a solid still persists and therefore still produces fluxes. Such a face either belongs to a valid or an orphaned cell. In the former case, it allocates its flux to its original valid cell. In the latter case, we know that the orphaned cell will be connected to a valid cell through a series of shared fluid faces. Therefore, any flux this face experiences represents some fluid that is able to make its way to the valid cell's particle via a path entirely within the fluid. As such, the face may allocate its flux to that valid cell, representing a physically viable flow.

Conversely, the presence of any solid face inhibits fluid flow by disconnecting the connectivity of either side of the face.

Thus, our partition explicitly preserves the connectivity of the fluid domain induced by the solid boundaries.

Because our partition preserves this connectivity, we produce a method that is sufficiently leakproof, but also leakproof only when necessary, allowing for fluid flux where the original solid boundary conditions allow for.

4.4 Numerical Flux

As noted in Equation 5, we require the evaluation of some numerical flux $\hat{\mathbf{F}}_{i,j}$ at each pairwise interface between neighbours i and j. Numerous methods exist to solve this, including an analytic solution via characteristic decomposition [Toro et al. 1994], and various linearization-based approximations [Einfeldt 1988; Harten et al. 1983; Roe 1981]. Our method is agnostic of this choice, and generally more accurate solvers exchange reduced diffusivity with increased computational cost.

In our implementation, we choose to take the numerical flux described in Kurganov and Tadmor [2000], which has been prior

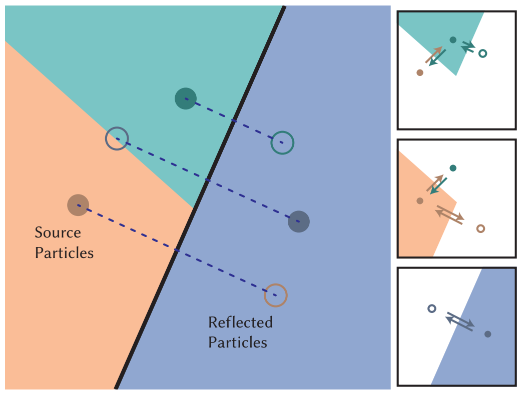


Fig. 7. Boundary condition enforcement via reflected particles. Reflected particles are depicted as outlined circles in the global state above, but only exist in the context of their corresponding particles in order to resolve the solid boundary condition (right). Note that locally, the solid interface is resolved exactly the same way as fluid interfaces, with a flux computed along the face.

shown to be sufficiently diffusive to reduce noise that would lead to multivalued fields. They use:

$$\hat{\mathbf{F}}_{ij} = \frac{1}{2}\left(\mathbf{F}^-_{ij} + \mathbf{F}^+_{ij}\right) - \frac{a_{ij}}{2}\left(\mathbf{U}^+_{ij} - \mathbf{U}^-_{ij}\right). \quad (6)$$

To define the signal velocity a_{ij}, consider a path $C\left(\mathbf{U}^-_{ij}, \mathbf{U}^+_{ij}\right)$, connecting the left and right states of a Riemann fan. The signal velocity is the spectral radius of the flux Jacobian $\partial\mathbf{F}/\partial\mathbf{U}$ evaluated along this curve. In other words, it is the largest magnitude eigenvalue along the path:

$$a_{ij} = \max_{\mathbf{U}\in C\left(\mathbf{U}^-_{ij}, \mathbf{U}^+_{ij}\right)} \lambda(\mathbf{U}). \quad (7)$$

For the compressible Euler equations, the eigenvalues of the flux Jacobian are $\lambda(\mathbf{U}) = \{u_n, u_n + c, u_n - c\}$. For a 1D Riemann problem, this is simply evaluates to the maximum eigenvalue at either side of the interface, $\max\left\{|u^-_n \pm c^-|, |u^+_n \pm c^+|\right\}$, with $c = \sqrt{\gamma P/\rho}$ being the sound speed of the fluid.

Conceptually, these numerical fluxes form waves traveling out from each interface, moving at the speed a_{ij}, as sketched out on Figure 6. Notice then that a_{ij} exactly determines the timestep restriction–waves cannot be allowed to travel past the middle of the cell. Outside this wave propagation domains, the solution remains the same from the previous timestep.

4.5 Boundary Enforcement

Because our discretization includes the solid boundaries as faces in the fluid partition, we greatly simplify the approach to respecting solid boundaries. We simply must enforce zero flux across the interface in the frame of the solid face's velocity, thus dictating that the fluid must match the solid face velocity at the interface.

To do so, for every solid face in a particle's fluid cell boundary, we create a reflected particle across this solid face, as shown on Figure 7. These reflected particles are entirely local, and exist only

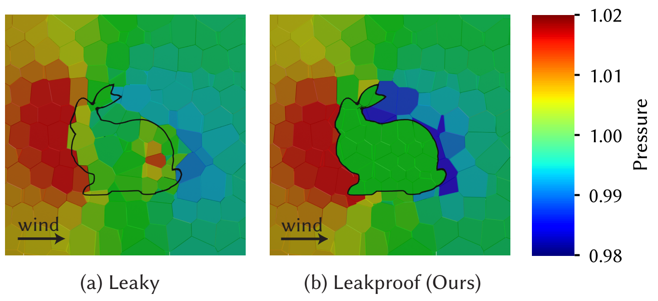


Fig. 8. Cutaway view of a bunny inside a windtunnel without (a) and with (b) our Voronoi stitching method. The fluid inside the watertight bunny remains quiescent using our method, but gains velocity via leaking through the interface using a naive approach.

from the point of view of the particle being reflected. They do not exist in the actual simulation domain, and do not participate in any other particle's flux computation. This differs from approaches with explicit solid Voronoi cells ([Hwang et al. 2021; Springel 2010]), which necessarily introduces a volume to the solid because of its inclusion in the Voronoi. We additionally note that this also allows us to resolve even subgrid boundaries immersed entirely within a cell, as those faces are still added to the mesh and enforce this fluid-solid boundary flux.

These reflected particles share the original particle's density and pressure, and will have a velocity mirrored in the face normal direction in the solid velocity's frame. Taking together the boost to solid velocity frame, performing the reflection, and deboosting to the lab frame, we have,

$$\mathbf{u}_f = \mathbf{u}_f - 2((\mathbf{u}_p - \mathbf{u}_s)\cdot\mathbf{n})\mathbf{n}, \quad (8)$$

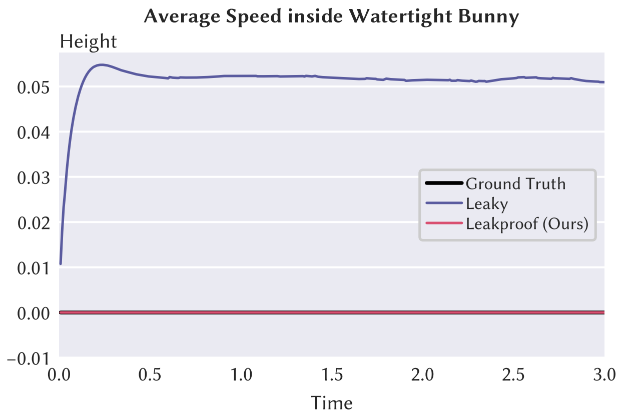


Fig. 9. Average speed over time inside a watertight bunny inside a windtunnel. Our leakproof partitioning keeps the interior quiescent for the duration of the simulation, matching the ground truth, while the naive approach gains 50% of the exterior velocity.

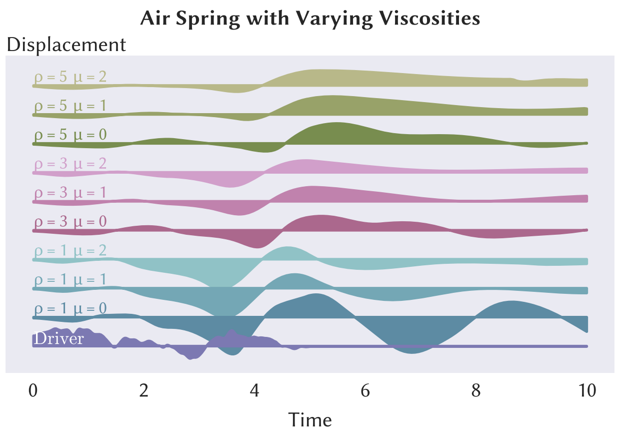


Fig. 10. Displacement over time in an airspring with a scripted driver input. Air densities and dynamic viscosities are as marked, with thermal conductivity taken to be $\kappa = 3\mu$.

with $\mathbf{u}_f$ being the fluid particle velocity, $\mathbf{u}_s$ being the solid face velocity, and $\mathbf{n}$ is the outwards-facing normal.

5 Results

We implemented our method as a solver node in Houdini [Side Effects Software Inc. 2025], using its built in rigid body and FEM solvers for solid simulation [Side Effects Software Inc. 2025]. Scene parameters are provided in Table 1.

5.1 Sealed Bunny

We place a static bunny boundary inside a wind tunnel, shown on Figure 8. The exterior fluid is initialized with a velocity of $u_x = 0.1$ lengthwise across the bunny, while the interior fluid is initialized with zero velocity. Inflow and outflow boundaries are placed at either end of the wind tunnel to maintain the exterior fluid velocity.

We simulate this watertight bunny boundary with both our leak-proof Voronoi stitching algorithm, and a naive algorithm of just using the existing Voronoi mesh with immersed solid boundary treatment. That is, the immersed solids still enforce the defined in Section 4.5. We find that our leakproof treatment retains the original fluid values up to machine single-precision, while the leaky bunny quickly gains velocity induced by the exterior wind, as shown on Figure 9.

As a stress test of our stitching algorithm, we also simulated this with the same exterior fluid particles, but seed only a single fluid particle on the bunny interior. We find that the stitching algorithm is able to correctly assign all cells inside the bunny to that single particle. This means the entire geometry is represented by a single particle's state. This state remained fully watertight and experienced no flux with the external fluid, thus retaining its original values up to machine precision.

5.2 Air Spring

Figure 10 shows the displacement from a simulated air spring, consisting of a sealed vertical chamber with two pistons constrained to vertical motion. The bottom piston is a scripted driver piston, while the top piston is allowed to freely move while experiencing gravity pulling it down and the pressure of the enclosed air pushing it up.

Demonstrating that our method can be expanded into viscous fluids, we introduce an extra viscous flux term to our numerical flux:

$$F^{visc} \cdot \mathbf{n} = \begin{bmatrix} 0 \\ \tau_{xj} n_j \\ \tau_{yj} n_j \\ \tau_{zj} n_j \\ u_i \tau_{ij} n_j - q_j n_j \end{bmatrix}, \tag{9}$$

where i, j are Einstein summation indices for three spatial dimensions. $\tau_{ij} = \mu \left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} - \frac{2}{3} \delta_{ij} \nabla \cdot \mathbf{u} \right)$ produces the viscous stress tensor with dynamic viscosity μ, and the heat flux is given by $q_j = -\kappa \frac{\partial T}{\partial x_j}$ with thermal conductivity κ. Temperature T was computed via ideal gas law $P/\rho = RT$, with R taken to be 1 in simulation units for the purpose of this example.

A render of a single travelling pulse is shown on Figure 11. We see that since sound speed decreases as density increases ($c_s = \sqrt{\gamma P/\rho}$), the follower piston takes longer to respond with increasing density. This has the added benefit of stretching out waves, thus smoothing out and reducing the follower's displacement. We also see that increasing viscosity has a similar effect, blending out sharp waves caused by sudden movements of the driver.

5.3 Stomp Rocket

A rocket mass is placed above an air-filled tube, connected to an air-filled reservoir through a u-bend. A driver "foot" is rapidly pushed down, sending air through the u-bend and into the reservoir, where

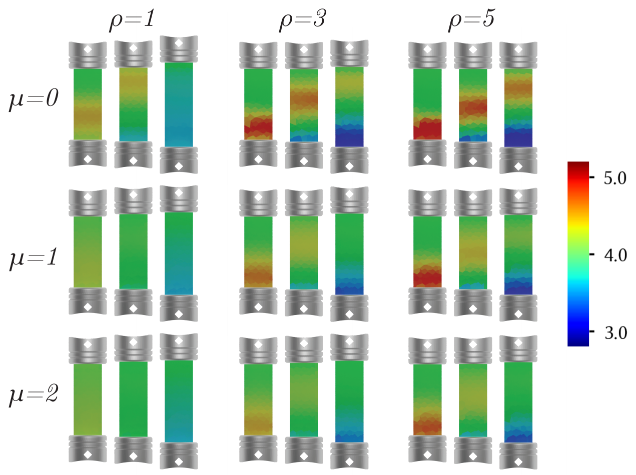


Fig. 11. Cutaway renders of pressure for airsprings of various densities and dynamic viscosities depicting a single propagating sine wave at $t = 0.4, 0.8, 1.2$.

Table 1. Scene parameters for presented examples. U_L and U_R represent the left and right states (or exterior and interior states, as prescribed in the problem) respectively, with ρ, u, P being fluid density, velocity, and pressure. Velocity is given as a scalar for the axis relevant to the problem, all other velocity components are 0. $\#P$ is the number of fluid particles and $\#V$ is the number of solid vertices. Machines used for computation were: an M1 Macbook Pro with 32GB RAM (Mac), an 8-core AMD R7 1700 CPU with 64GB RAM (PC), and a 32-core AMD Threadripper 2990WX with 128 GB of RAM (Server).

	U_L			U_R						*Fluid*	*Solid*		
	ρ	u	P	ρ	u	P	*Domain*	*#Frames*	*Time*	*#P*	*#V*	*Machine*	*s/Frame*
Sealed Bunny													
Leakproof	1	0.1	1	1	0	1	2 x 2 x 2	360	3	2652	6938 Static	Mac	29.67
Leaky	1	0.1	1	1	0	1	2 x 2 x 2	360	3	2652	6948 Static	Mac	24.00
Air Spring	Var.	0	4	-	-	-	1 x 9 x 1	7200	15	672	263 Scripted	Server	1.74
											263 Rigid		
											433 Static		
Stomp Rocket	Var.	0	4	-	-	-	3 x 4 x 2	1200	5	955	262 Scripted	Server	0.85
											258 Rigid		
											816 Static		
Asteroid	1	Var.	1	-	-	-	1 x 1 x 1	1920	2	22336	674 Rigid	Server	135.91
Champagne	1	0	1	10	0	10	1 x 1 x 1	1920	1	732	463 Rigid	PC	30.94
											8382 Static		
Balloon	1	0	6	-	-	-	10 x 10 x 10	4800	10	1350	824 Deformable	PC	8.25
Fan													
Square Duct	1	0	20	-	-	-	1.5 x 1.5 x 5	4800	20	5512	2395 Scripted	Server	16.32
Circular Duct	1	0	20	-	-	-	1.5 x 1.5 x 5	4800	20	3703	2395 Scripted	Server	16.25
											2391 Static		

it pushes the rocket out; we plot height of the rocket over time on Figure 12. We test a variety of sound speed by modulating the density of the internal fluid, and see that fluids with lower sound speeds achieve higher overall heights.

Although the trajectory of the foot is the same for all tests, the work done is larger for lower sound speeds. As sound speed decreases, the propagating wave moves slower, therefore more gas "piles" up, increasing pressure and therefore total work done. This increased work translates to faster upwards velocity on the rocket. On the opposite limit is the incompressible case, where the rocket will only travel the same distance as the foot travels down.

5.4 Asteroid

To demonstrate that our method handles both subsonic and supersonic flow around solids, we simulate fluid flow around Asteroid 4486 Mithra [NASA Science 2025] Three different flow speeds and their corresponding mach numbers are shown on Figure 13, with the ground truth Mach cones shown as white lines.

We note one deficiency with our current method is the loss of resolution in the supersonic wake. Because the fluid particles move with the velocity, the wakes tend to become underresolved. This is especially problematic for highly supersonic flow, such as the rightmost case, where no particles at all are found in the wake. Our Voronoi discretization fills this in with the high pressure data from the neighbouring particles, improperly representing the low pressure wake. This may be resolved with careful particle splitting whenever particle density becomes too low.

5.5 Champagne Cork

We simulate the ejection of a champagne cork from a bottle, where internal pressure gradually increases, representing dissolved CO_2 bubbling out of the solution. The cork is initially held in place by friction between the cork and the neck, which is overcome by the increasing pressure of the air pocket, after which the cork is released and shot out of the bottle. Our method seamlessly handles the topology change as the interior domain merges with the exterior domain, allowing fluid to start flowing outwards.

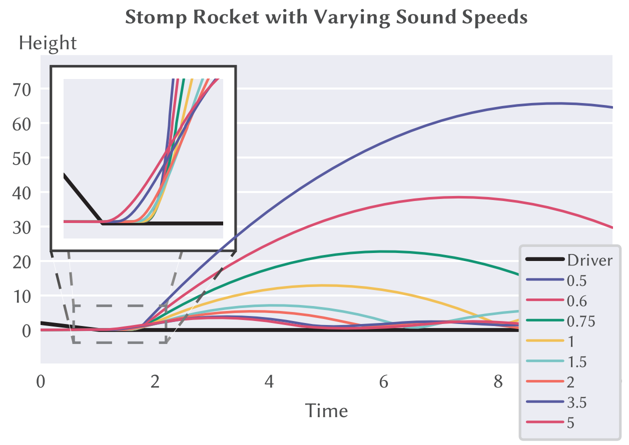


Fig. 12. Displacement over time of a stomp rocket with a zoom in of early time inset.

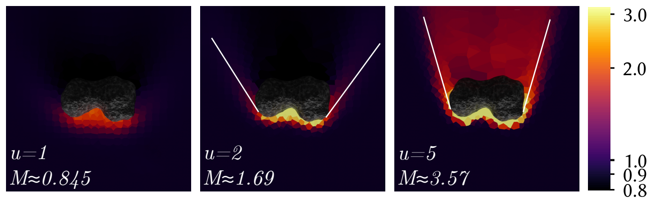


Fig. 13. Cutaway view of pressure around an asteroid falling at various speeds, with the corresponding Mach number as marked. Ground truth Mach cones are shown as white lines for the two supersonic cases.

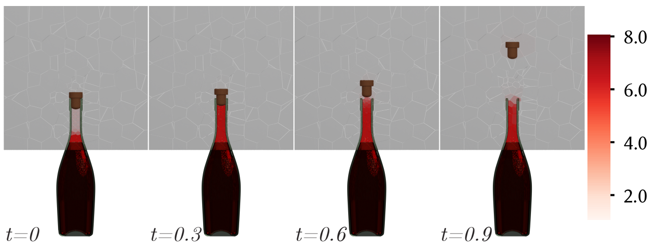


Fig. 14. Cutaway view of pressure in a champagne bottle launching a cork held on by friction. Pressure inside the bottle is initially at 2 atm, and slowly rises to 10 atm via an inflow boundary at the wine surface. Outer domain consists of outflow boundary conditions.

5.6 Balloon Propulsion

We allow an inflated balloon to release its stored air, propelling itself through the simulation domain, as shown on Figure 1. This demonstrates complex energy transfer between stored potential energy of the elastic balloon, potential energy of high pressure air, into fluid kinetic energy as air escapes the balloon due to the pressure difference inside and outside the balloon, to kinetic energy of the balloon as flies around the domain.

5.7 Fan

We push air using a scripted rotating fan along two different duct geometries, shown on Figure 15. We note that this example relies heavily on the resolution of subgrid surfaces, as the fan is considerably thinner than the particle resolution. Even here, our method is able to transfer velocity from the solid to the fluid phase.

We see that the circular duct is able to more efficiently accelerate air, as shown by the greater pressure gradient across the fan, with the square duct demonstrating a considerable amount of backflow at the corners, shown by the negative axial velocities.

6 Discussion and Conclusions

We presented a method for coupling compressible fluids with solids while respecting the connectivity of the fluid, creating a discretization that is leakproof only when required, and flow permissive otherwise. Our method leverages the Voronoi diagram to partition space, modifying it to conform to solid boundaries and allow for natural handling of force transfer between the solid and fluid regimes.

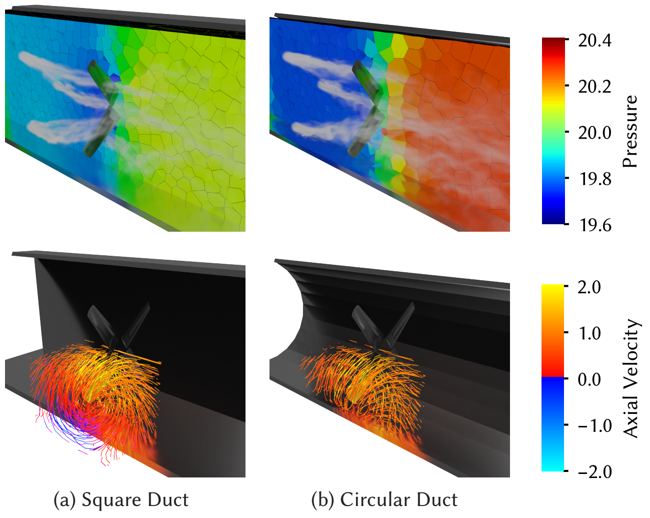


Fig. 15. Cutaway view of the pressure gradient induced as a rotating fan pushes air forwards (above), and streamlines coloured by the axial flow velocity (below), for two different duct geometries. Smoke was generated in post-processing, using our computed velocity field.

We identify extensions of our work that are ripe avenues for future research. In particular, because of the explicit nature of our method, flux computation and advection are extremely fast and embarrassingly parallelizable. Our method, however, is significantly bottlenecked by the cost of computing the Voronoi diagram. Our current implementation uses Houdini's built-in library, which recomputes the Voronoi structure at each timestep, taking up 85% of our simulation runtime. Leveraging temporal coherence to update the previous timestep's structure is a promising method for accelerating [Guibas 2018]. We, however, note that the diagram is only required for connectivity and face areas. If there was a method for approximating face areas while maintaining the same connectivity properties without fully computing a Voronoi diagram, the method may become significantly faster.

Additionally, the modifications to the Voronoi does sidestep the true intended structure, which is a visibility-constrained Voronoi. That is, every cell should have the domain of every point in space closest to it, while accounting for visibility constraints (the solid boundaries). Imagine the source points growing in the domain given some occluding planes, the points would propagate out waves that naturally bend around the ends of the occluding planes. Some work exists for rectilinear occluders in 2D [Tsin and Wang 1996], but extensions to arbitrary barriers in 3D remain an open problem. Similar work can also be used for more accurate intracell gradients, as gradients to faces should vary according to occluders within these cells.

We also found that in certain examples, such as the supersonic wake in Section 5.4, the Lagrangian particles become underresolved. This could be alleviated via a particle splitting scheme wherever local particle density becomes too low. We emphasize, however, that our method still maintains our desired connectivity property even when resolution drops like this, with the neighbouring particles simply extending their domains into the underresolved regions.

Finally, because of our choice of explicit method, coupling is handled by the multiple timestepping. That is, fluid imposes some boundary condition on the solid, which evolves and imposes a boundary condition onto the liquid. A more accurate approach would be to find the force balance between the two regimes *within* each timestep, such as with a Newton iterative solver.

Research on compressible flow is typically driven by applications in science and engineering, some of which fall under ethically grey domains. Beyond technical novelty, we hope that our work highlights the rich visual and physical phenomena that emerge from advancing algorithms in comppressible flow, and that it encourages further exploration in creative and constructive directions.

References

Franz Aurenhammer, Therese Biedl, Michel Brevilliers, Thomas Hackl, Rolf Klein, Elmar Langetepe, and Günter Rote. 2014. A note on visibility-constrained Voronoi diagrams. *Inform. Process. Lett.* 114, 7 (2014), 357–361. https://doi.org/10.1016/j.ipl.2014.01.014

Markus Becker and Matthias Teschner. 2007. Weakly compressible SPH for free surface flows. In *Proceedings of the 2007 ACM SIGGRAPH/Eurographics symposium on Computer animation.* 209–217.

Jan Bender and Dan Koschier. 2016. Divergence-free SPH for incompressible and viscous fluids. *IEEE Transactions on Visualization and Computer Graphics* 23, 3 (2016), 1193–1206.

Marsha J Berger and Phillip Colella. 1989. Local adaptive mesh refinement for shock hydrodynamics. *Journal of computational Physics* 82, 1 (1989), 64–84.

Christoph Börgers and Charles S Peskin. 2005. A Lagrangian method based on the Voronoi diagram for the incompressible Navier Stokes equations on a periodic domain. In *The Free-Lagrange Method: Proceedings of the First International Conference on Free-Lagrange Methods, Held at Hilton Head Island, South Carolina, March 4–6, 1985*. Springer, 87–113.

Robert Bridson. 2015. *Fluid simulation for computer graphics.* AK Peters/CRC Press.

Tyson Brochu, Christopher Batty, and Robert Bridson. 2010. Matching fluid simulation elements to surface geometry and topology. In *ACM SIGGRAPH 2010 papers.* 1–9.

Oleksiy Busaryev, Tamal K Dey, Huamin Wang, and Zhong Ren. 2012. Animating bubble interactions in a liquid foam. *ACM Transactions on Graphics (TOG)* 31, 4 (2012), 1–8.

Yadi Cao, Yunuo Chen, Minchen Li, Yin Yang, Xinxin Zhang, Mridul Aanjaneya, and Chenfanfu Jiang. 2022. An efficient b-spline lagrangian/eulerian method for compressible flow, shock waves, and fracturing solids. *ACM Transactions on Graphics (TOG)* 41, 5 (2022), 1–13.

M Desbrun. 1996. Smoothed particles: A new paradigm for animating highly deformable bodies. *Computer Animation and Simulation/Springer Vienna* (1996).

Jean Donea, SHJP Giuliani, and Jean-Pierre Halleux. 1982. An arbitrary Lagrangian-Eulerian finite element method for transient dynamic fluid-structure interactions. *Computer methods in applied mechanics and engineering* 33, 1-3 (1982), 689–723.

Bernd Einfeldt. 1988. On Godunov-type methods for gas dynamics. *SIAM Journal on numerical analysis* 25, 2 (1988), 294–318.

Ronald P Fedkiw. 2002. Coupling an Eulerian fluid calculation to a Lagrangian solid calculation with the ghost fluid method. *J. Comput. Phys.* 175, 1 (2002), 200–224.

Ronald P Fedkiw, Tariq Aslam, Barry Merriman, and Stanley Osher. 1999. A non-oscillatory Eulerian approach to interfaces in multimaterial flows (the ghost fluid method). *Journal of computational physics* 152, 2 (1999), 457–492.

Jeferson Wilian Dossa Fernandes, Humberto Breves Coda, and Rodolfo André Kuche Sanches. 2019. ALE incompressible fluid–shell coupling based on a higher-order auxiliary mesh and positional shell finite element. *Computational Mechanics* 63 (2019), 555–569.

Reza Ghias, Rajat Mittal, and Haibo Dong. 2007. A sharp interface immersed boundary method for compressible viscous flows. *J. Comput. Phys.* 225, 1 (2007), 528–553.

Robert A Gingold and Joseph J Monaghan. 1977. Smoothed particle hydrodynamics: theory and application to non-spherical stars. *Monthly notices of the royal astronomical society* 181, 3 (1977), 375–389.

Jón Tómas Grétarsson and Ron Fedkiw. 2013. Fully conservative leak-proof treatment of thin solid structures immersed in compressible fluids. *J. Comput. Phys.* 245 (2013), 160–204.

Leonidas Guibas. 2018. Kinetic data structures. In *Handbook of Data Structures and Applications*. Chapman and Hall/CRC, 377–388.

Amiram Harten, Peter D Lax, and Bram van Leer. 1983. On upstream differencing and Godunov-type schemes for hyperbolic conservation laws. *SIAM review* 25, 1 (1983), 35–61.

Xiaowei He, Shusen Liu, Yuzhong Guo, Jian Shi, and Ying Qiao. 2025. A Semi-Implicit SPH Method for Compressible and Incompressible Flows with Improved Convergence. In *Computer Graphics Forum*. Wiley Online Library, e70043.

Cyrill W Hirt, Anthony A Amsden, and JL Cook. 1974. An arbitrary Lagrangian-Eulerian computing method for all flow speeds. *Journal of computational physics* 14, 3 (1974), 227–253.

Philip F Hopkins. 2015. A new class of accurate, mesh-free hydrodynamic simulation methods. *Monthly Notices of the Royal Astronomical Society* 450, 1 (2015), 53–110.

Young Kwang Hwang, John E Bolander, Yun Mook Lim, and Jung-Wuk Hong. 2021. Coupling of SPH and Voronoi-cell lattice models for simulating fluid–structure interaction. *Computational Particle Mechanics* 8, 4 (2021), 813–823.

Alexander Kurganov and Eitan Tadmor. 2000. New high-resolution central schemes for nonlinear conservation laws and convection–diffusion equations. *Journal of computational physics* 160, 1 (2000), 241–282.

Nathalie Lanson and Jean-Paul Vila. 2008. Renormalized meshfree schemes I: consistency, stability, and hybrid methods for conservation laws. *SIAM J. Numer. Anal.* 46, 4 (2008), 1912–1934.

Peter D. Lax. 1954. Weak solutions of nonlinear hyperbolic equations and their numerical computation. *Communications on Pure and Applied Mathematics* 7, 1 (1954), 159–193.

Leon B Lucy. 1977. A numerical approach to the testing of the fission hypothesis. *Astronomical Journal, vol. 82, Dec. 1977, p. 1013-1024.* 82 (1977), 1013–1024.

DJ Mavriplis and V Venkatakrishnan. 1997. A unified multigrid solver for the Navier-Stokes equations on mixed element meshes. *International Journal of Computational Fluid Dynamics* 8, 4 (1997), 247–263.

Rajat Mittal, Haibo Dong, Meliha Bozkurttas, FM Najjar, Abel Vargas, and Alfred Von Loebbecke. 2008. A versatile sharp interface immersed boundary method for incompressible flows with complex boundaries. *Journal of computational physics* 227, 10 (2008), 4825–4852.

NASA Science. 2025. Asteroid 4486 Mithra 3D Model. https://science.nasa.gov/3d-resources/asteroid-4486-mithra/ Accessed: 2025-05-20.

Atsuyuki Okabe, Barry Boots, Kokichi Sugihara, and Sung Nok Chiu. 2000. *Spatial Tessellations: Concepts and Applications of Voronoi Diagrams* (2nd ed.). John Wiley & Sons, Chichester.

Charles S Peskin. 1972. Flow patterns around heart valves: a numerical method. *Journal of computational physics* 10, 2 (1972), 252–271.

Charles S Peskin. 2002. The immersed boundary method. *Acta numerica* 11 (2002), 479–517.

Ziyin Qu, Minchen Li, Yin Yang, Chenfanfu Jiang, and Fernando De Goes. 2023. Power Plastics: A Hybrid Lagrangian/Eulerian Solver for Mesoscale Inelastic Flows. *ACM Transactions on Graphics (TOG)* 42, 6 (2023), 1–11.

Phil W Randles and Larry D Libersky. 1996. Smoothed particle hydrodynamics: some recent improvements and applications. *Computer methods in applied mechanics and engineering* 139, 1-4 (1996), 375–408.

Philip L Roe. 1981. Approximate Riemann solvers, parameter vectors, and difference schemes. *Journal of computational physics* 43, 2 (1981), 357–372.

Robert I Saye and James A Sethian. 2011. The Voronoi implicit interface method for computing multiphase physics. *Proceedings of the National Academy of Sciences* 108, 49 (2011), 19498–19503.

Hagit Schechter and Robert Bridson. 2012. Ghost SPH for animating water. *ACM Transactions on Graphics (TOG)* 31, 4 (2012), 1–8.

Mar Serrano, Pep Español, and Ignacio Zúñiga. 2005. Voronoi fluid particle model for Euler equations. *Journal of statistical physics* 121 (2005), 133–147.

Side Effects Software Inc. 2025. *Houdini.* https://www.sidefx.com/ 3D Animation and Visual Effects Software.

Funshing Sin, Adam W Bargteil, and Jessica K Hodgins. 2009. A point-based method for animating incompressible flow. In *Proceedings of the 2009 ACM SIGGRAPH/Eurographics symposium on computer animation.* 247–255.

Volker Springel. 2010. E pur si muove: Galilean-invariant cosmological hydrodynamical simulations on a moving mesh. *Monthly Notices of the Royal Astronomical Society* 401, 2 (2010), 791–851.

James M Stone and Michael L Norman. 1992. ZEUS-2D: a radiation magnetohydrodynamics code for astrophysical flows in two space dimensions. I-The hydrodynamic algorithms and tests. *Astrophysical Journal Supplement Series (ISSN 0067-0049), vol. 80, no. 2, June 1992, p. 753-790. Research supported by University of Illinois.* 80 (1992), 753–790.

Hidenori Takeda, Shoken M Miyama, and Minoru Sekiya. 1994. Numerical simulation of viscous flow by smoothed particle hydrodynamics. *Progress of theoretical physics* 92, 5 (1994), 939–960.

Eleuterio F Toro. 2013. *Riemann solvers and numerical methods for fluid dynamics: a practical introduction.* Springer Science & Business Media.

Eleuterio F Toro, Michael Spruce, and William Speares. 1994. Restoration of the contact surface in the HLL-Riemann solver. *Shock waves* 4 (1994), 25–34.

Yung H. Tsin and Cao An Wang. 1996. Geodesic Voronoi diagrams in the presence of rectilinear barriers. *Nord. J. Comput.* 3, 1 (1996), 1–26.

Vuko Vukčević, Hrvoje Jasak, and Inno Gatin. 2017. Implementation of the ghost fluid method for free surface flows in polyhedral finite volume framework. *Computers & fluids* 153 (2017), 1–19.

Tao Ye, Rajat Mittal, HS Udaykumar, and Wei Shyy. 1999. An accurate Cartesian grid method for viscous incompressible flows with complex immersed boundaries. *Journal of computational physics* 156, 2 (1999), 209–240.