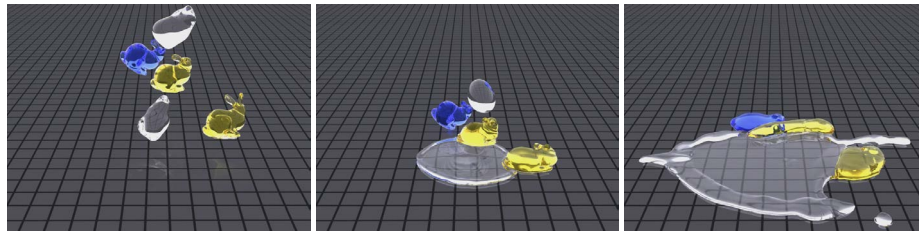


# Multiphase Flow of Immiscible Fluids on Unstructured Moving Meshes

M. K. Misztal<sup>1</sup>, K. Erleben<sup>2</sup>, A. Bargteil<sup>3</sup>, J. Fursund<sup>4</sup>, B. Bunch Christensen<sup>4</sup>, J. A. Bærentzen<sup>1</sup> and R. Bridson<sup>5</sup>

<sup>1</sup>Dept. Informatics and Mathematical Modelling, Technical University of Denmark,  
<sup>2</sup>Dept. of Computer Science, University of Copenhagen, Denmark, <sup>3</sup>School of Computing, University of Utah, USA,  
<sup>4</sup>Alexandra Institute, Denmark, <sup>5</sup>Dept. Computer Science, University of British Columbia, Canada



**Figure 1:** Multiple fluids with different viscosity coefficients and surface tension densities splashing on the bottom of a cylindrical container. Observe that the simulation has no problem dealing with thin sheets.

## Abstract

In this paper, we present a method for animating multiphase flow of immiscible fluids using unstructured moving meshes. Our underlying discretization is an unstructured tetrahedral mesh, the deformable simplicial complex (DSC), that moves with the flow in a Lagrangian manner. Mesh optimization operations improve element quality and avoid element inversion. In the context of multiphase flow, we guarantee that every element is occupied by a single fluid and, consequently, the interface between fluids is represented by a set of faces in the simplicial complex. This approach ensures that the underlying discretization matches the physics and avoids the additional book-keeping required in grid-based methods where multiple fluids may occupy the same cell. Our Lagrangian approach naturally leads us to adopt a finite element approach to simulation, in contrast to the finite volume approaches adopted by a majority of fluid simulation techniques that use tetrahedral meshes. We characterize fluid simulation as an optimization problem allowing for full coupling of the pressure and velocity fields and the incorporation of a second-order surface energy. We introduce a preconditioner based on the diagonal Schur complement and solve our optimization on the GPU. We provide the results of parameter studies as well as a performance analysis of our method.

Categories and Subject Descriptors (according to ACM CCS): Computer Graphics [I.3.5]: Computational Geometry and Object Modeling—Physically-based modeling Computer Graphics [I.3.7]: Three-Dimensional Graphics and Realism—Animation Mathematics of Computing [G.1.6]: Optimization—Nonlinear programming

## 1. Introduction

In this paper, we present a finite element method for animating multiphase flow of immiscible fluids. Our approach is based on the Lagrangian deformable simplicial complex (DSC) method, previously used for topology-adaptive de-

formable interface tracking [Mis10,MB12], which trades the apparent simplicity of the level set method for robustness and support of multiple phases, as well as offering an unstructured, moving computational grid. Each element in the mesh is assigned a single material and interfaces are com-

posed of faces in the DSC. The DSC moves with the fluid in a Lagrangian fashion and a variety of mesh optimization operations improve element quality and avoid element inversion. We formulate the solution of the Navier-Stokes equations in terms of a quadratic optimization problem, which accounts for and couples all terms: incompressibility, viscosity, surface energy and arbitrary solid constraints.

Our approach builds on the work of Misztal et al. [MBE\*10] and Erleben et al. [EMB11], who developed a finite element approach to fluid simulation and characterized the solution to the Navier-Stokes equations as a quadratic optimization problem. Our main contribution is the extension of these techniques to multiphase flow. Furthermore, we have addressed numerous implementation issues, essential for significant improvement of the accuracy of the method and reduction of its time complexity. Our contributions include:

- deriving the formulation of and implementing the second-order surface energy approximation (Section 3.3);
- deriving and implementing pressure stabilization through finite volume discretization of the pseudo-compressibility equation (in contrast to the previously used pressure stabilization scheme from Misztal et al. [MBE\*10], used by Erleben et al. [EMB11], which is not physically correct; see Section 3.4);
- designing a preconditioner, which allows us to employ a GPU-based, iterative solver (Section 3.5);
- simplifying the formulation of the solid boundary conditions in the model (Section 3.6);
- adapting the method so that it supports multiple, immiscible fluids (Section 3.7).

In contrast to regular grid and level set based approaches, our method offers several significant advantages: the Lagrangian nature of our mesh avoids numerical diffusion that leads to volume loss and excessive perceived viscosity; the unstructured tetrahedral mesh allows us to trivially handle arbitrary, non-grid-aligned solid boundaries, and the explicit representation of interfaces as faces in the mesh allows for accurate treatment of surface tension. In the context of multiphase flow we wish to highlight one additional advantage: the Lagrangian nature of our discretization allows us to optimally track the interfaces between multiple fluids. There is no guesswork in determining what fluids are where and the simulation is greatly simplified because each element is occupied by a single material. This also allows us to assign different values of surface energy density to all pairs of materials. We present several examples of fluid simulations generated using our method, as well as the results of various performance tests and parameter studies.

## 2. Related Works

The literature concerning fluid simulation in computer graphics is rich, and a proper review of all the state-of-the-art methods would require a study of its own. Most methods

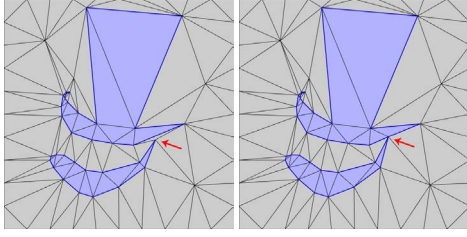
are based on regular grids and we refer to [FM96, FM97, Sta99, FF01, FSJ01, BMF07, Bri08, BBB10, MST10] for details. The literature on multiphase flow is sparser. Losasso et al. [LSSF06] first demonstrated multiple interacting fluids. They used multiple particle level sets to track the various interfaces and introduced averaging rules for handling cells occupied by more than one material. Kim [Kim10] expanded on this approach by introducing regional level sets [ZYP06].

Another big group of methods is based on smoothed-particle hydrodynamics [Mon92] and the work by Solenthaler et al. [SSP07] is of particular interest here since it is able to handle multiple materials robustly. In contrast, our method uses an unstructured grid. The work in computer graphics on fluid solvers based on unstructured meshes is sparse. Early work used static unstructured meshes [FOK05, ETK\*07]. Dynamic meshes with limited deformation (to preserve mesh quality) were demonstrated by Feldman et al [FOKG05]. When mesh quality can no longer be preserved, an entirely new mesh can be generated [KFCO06, BWHT07, CFL\*07, WT08] though this involves a great deal of computation and can lead to undesirable artifacts, such as the smoothing of simulation variables. Alternatively, local mesh improvement operations [MBE\*10, WRK\*10] are computationally more efficient and minimize artifacts. Solid boundaries and two-way coupling have been touched upon [FOK05, KFCO06]. The preferred method for dealing with advection has been the semi-Lagrangian advection method [FOK05] and its generalization to deforming meshes [FOKG05] which has been applied in many works [KFCO06, CGFO06, CFL\*07, WBOL07].

The finite volume method is a popular choice for fluid simulation on unstructured meshes [FOK05, FOKG05, KFCO06, ETK\*07, WBOL07]. In particular, Elcott et al. [ETK\*07] demonstrate a number of desirable, even surprising, properties for incompressible flow simulation with their use of discrete exterior calculus. In contrast, the finite element method [BWHT07, WT08, WRK\*10] is often preferred for plastic and elastic objects. However, its application for fluids is sparse [MBE\*10, EMB11]. Mimicking regular grids, many tetrahedral schemes are based on staggered locations of simulation variables [FOK05, FOKG05, ETK\*07, WBOL07]. Here the face centers often store the normal velocities and volume centers store pressure values. While this approach has a number of nice properties [ETK\*07], they have the significant drawback that reconstruction of the full-dimensional velocity field is quite expensive. In contrast, while we do store pressures at the centers of our elements, the velocity field is stored as full-dimensional velocity vectors at the nodes of our mesh.

## 3. Fluid Simulation Method

In this section we present the complete, finite element discretization of the incompressible fluid motion equations and its formulation in terms of a quadratic optimization problem,

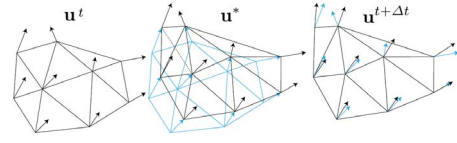


**Figure 2:** A 2D example of an interface representation in the DSC method. The embedding mesh can be re-tessellated in order to accommodate vertex displacement and produce changes in the topology of the interface.

which allows us to accurately incorporate non-linear terms (such as surface tension forces) into the model. We also discuss the significant implementation details required for a robust performance: pressure stabilization (in the form of pseudo-compressibility) and preconditioning, which allows us to improve the performance by employing a fast, GPU-based, iterative solver. Finally, we describe how to handle the fluid’s interactions with arbitrary solid walls, and how to adapt the method to handle several interacting, immiscible fluids.

### 3.1. The Deformable Simplicial Complex Method

Our FEM computations are performed on an unstructured, tetrahedral grid, which is also used by the *deformable simplicial complex* (DSC) method [Mis10, MB12] for tracking the fluid’s free surface. The DSC method is a recent, Lagrangian method for topology-adaptive deformable interface tracking. It represents the interface explicitly as a piecewise surface (triangle mesh), while the whole embedding space is discretized as well – as a tetrahedral mesh. All tetrahedra are labeled *inside* or *outside* according to their location relative to the interface (the 2D case is shown in Figure 2). Furthermore, they conform to the interface, in the sense that each interface triangle is a common face shared by one *inside* tetrahedron and one *outside* tetrahedron. The interface deformations are produced by iterating interface vertex displacement according to a given velocity field, followed by a mesh improvement step. The main purpose of the mesh improvement step is the removal of low quality tetrahedra produced during vertex advection and reducing the risk of creating inverted tetrahedra in subsequent deformation steps. It is inspired by earlier works on tetrahedral mesh improvement [FOG97, KS07] and consists of Laplacian and optimization-based smoothing of non-interface vertices, mesh reconnection operations, vertex insertion through edge splitting, edge collapse and removal of degenerate elements. Furthermore, the DSC method allows for surface mesh improvements through edge flips and *null-space smoothing* [Jia07]. For the purpose of further improving the computational mesh quality, we have augmented



**Figure 3:** Schematic view of a single iteration of our fluid simulation method (elements on the outside of the fluid are not shown for the sake of clarity). We begin with an initial velocity field  $\mathbf{u}^t$  respecting the continuity constraint (left). Then we move the grid according to this velocity field, advect the velocity values and possibly re-tessellate the mesh (dotted line) in order to accommodate the displacements or improve its quality (center). The new velocity field  $\mathbf{u}^*$  might violate the continuity constraint. In order to fix that, we solve the discretized version of the Poisson equation and obtain the final velocity field  $\mathbf{u}^{t+\Delta t}$  (right).

the DSC method with an optimization-based vertex insertion algorithm, based on the work by Klingner and Shewchuk [KS07].

The main advantages of the DSC method in the context of fluid simulation include: robust topological adaptivity, low numerical diffusion, available surface mesh representation, which does not change gratuitously between time steps, and the possibility of representing more than two phases (one can use an arbitrary number of tetrahedron labels rather than just two).

### 3.2. Fluid Simulation as a Quadratic Optimization Problem

We treat the tetrahedra contained in each fluid volume as conforming, linear elements. We sample the velocity field  $\mathbf{u}(x, y, z)$  at each vertex of the mesh  $\mathbf{x}_i$ ,  $i = 1, \dots, N_V$ , where  $N_V$  is the total number of vertices in the mesh. We denote the vector of all velocity samples as  $\mathbf{u} = [\mathbf{u}_1^T \ \mathbf{u}_2^T \ \dots \ \mathbf{u}_{N_V}^T]^T$ , where  $\mathbf{u}_i = \mathbf{u}(\mathbf{x}_i)$ . We are using a *staggered* grid, meaning that the pressure field is discretized at tetrahedra:  $\mathbf{p} = [p_1 \ p_2 \ \dots \ p_{N_T}]^T$ , where  $N_T$  is the number of elements (tetrahedra) occupied by the fluid. The optimization-based fluid simulation method is a fractional step method. In the first step we perform (forward Euler) vertex advection according to the current sampled velocity field  $\mathbf{u}^t$

$$\mathbf{x}_i^{t+\Delta t} = \mathbf{x}_i^t + \mathbf{u}_i^t \Delta t \quad (1)$$

and we advect the velocity values along with vertices obtaining an intermediate velocity field  $\mathbf{u}^*$  which might violate the new, discretized continuity constraint. We fix that in the second step by solving the finite-element discretization of the fluid motion equations in the form of an optimization problem, which determines the final velocity field  $\mathbf{u}^{t+\Delta t}$  of the

fluid (the details of the discretization are presented in Section 3.3)

$$\mathbf{u}^{t+\Delta t} = \arg \min_{\mathbf{u}} \frac{1}{2} \mathbf{u}^T \mathbf{A} \mathbf{u} + \mathbf{u}^T \mathbf{b} \quad (2)$$

subject to

$$\mathbf{P}^T \mathbf{u} = \mathbf{0} \quad (3)$$

where  $\mathbf{A}$  accounts for inertia, viscosity and surface tension,  $\mathbf{b}$  contains the effect of the advection and external force densities like gravity, and  $\mathbf{P}$  is the gradient operator. The Karush–Kuhn–Tucker (KKT) conditions for this problem read

$$\underbrace{\begin{bmatrix} \mathbf{A} & \mathbf{P} \\ \mathbf{P}^T & \mathbf{0} \end{bmatrix}}_{\mathbf{K}} \begin{bmatrix} \mathbf{u} \\ \lambda \end{bmatrix} = \begin{bmatrix} -\mathbf{b} \\ \mathbf{0} \end{bmatrix} \quad (4)$$

where  $\lambda$  are the Lagrange multipliers and correspond to  $-\Delta t \mathbf{p}$  — the pressure field multiplied by the time step size. Observe that solving this problem fully couples the velocity and pressure fields, unlike the projection method. The  $\mathbf{K}$  matrix is named the KKT matrix and is known to be symmetric indefinite [NW99].

### 3.3. Navier-Stokes Equation Discretization

The motion of a Newtonian fluid is governed by the Navier-Stokes equation,

$$\rho \dot{\mathbf{u}} = -(\mathbf{u} \cdot \nabla) \mathbf{u} + \nabla \cdot \mathbf{T} + \mathbf{f}, \quad \mathbf{x} \in \mathcal{V}_{\text{fluid}} \quad (5)$$

where  $\mathcal{V}_{\text{fluid}} \subset \mathbf{R}^3$  is the volume of the fluid,  $\rho$  is the mass density,  $\mathbf{u}$  is the unknown velocity field, and  $\mathbf{T}$  is the Newtonian stress tensor:

$$\mathbf{T} = -p \mathbf{I}_{3 \times 3} + \mu (\nabla \mathbf{u} + \nabla \mathbf{u}^T), \quad (6)$$

where  $p$  is the pressure field,  $\mu$  is the dynamic viscosity coefficient, and  $\mathbf{f}$  is an external force term (for example gravity). We assume constant mass density which yields a continuity constraint in the form of incompressibility,

$$\nabla \cdot \mathbf{u} = 0, \quad \mathbf{x} \in \mathcal{V}_{\text{fluid}}. \quad (7)$$

Erleben et al. [EMB11] show that the weak formulation of this system for a tetrahedral mesh with a staggered grid layout is as follows:

$$\mathbf{M} \frac{\partial \mathbf{u}}{\partial t} - \mathbf{B} \mathbf{f} - \mathbf{P} \mathbf{p} + \mathbf{D} \mathbf{u} = \mathbf{0}, \quad (8)$$

$$\mathbf{P}^T \mathbf{u} = \mathbf{0}, \quad (9)$$

where  $\mathbf{f} = [\mathbf{f}_1^T \quad \mathbf{f}_2^T \quad \dots \quad \mathbf{f}_{N_V}^T]^T$  and

$$\mathbf{M}_{ij} = \mathbf{I}_{3 \times 3} \int_{\mathcal{V}_{\text{fluid}}} \rho \phi_i \phi_j dV, \quad (10)$$

$$\mathbf{B}_{ij} = \mathbf{I}_{3 \times 3} \int_{\mathcal{V}_{\text{fluid}}} \phi_i \phi_j dV, \quad (11)$$

$$\mathbf{D}_{ij} = \int_{\mathcal{V}_{\text{fluid}}} \mu (\nabla \phi_i^T \nabla \phi_j \mathbf{I}_{3 \times 3} + \nabla \phi_i \nabla \phi_j^T) dV, \quad (12)$$

$$\mathbf{P}_{jk} = \int_{\mathcal{V}_k} \nabla \phi_j dV, \quad (13)$$

where  $i, j = 1, 2, \dots, N_V$ ,  $k = 1, 2, \dots, N_T$  and  $\mathcal{V}_k$  is the volume of the  $k^{\text{th}}$  tetrahedron. The shape functions  $\phi_i: \mathbf{R}^3 \mapsto \mathbf{R}$  fulfill the condition

$$\phi_i(\mathbf{x}_j) = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{if } i \neq j \end{cases} \quad (14)$$

and are piecewise linear over each element, which allows us to evaluate the matrices above analytically. We apply the finite difference method to discretize Equation 8, by substituting  $\frac{\partial \mathbf{u}}{\partial t} \approx \frac{1}{\Delta t} (\mathbf{u}^{t+\Delta t} - \mathbf{u}^*)$  and, by choosing an implicit scheme for stability we obtain the following system of linear equations

$$\mathbf{A} \mathbf{u}^{t+\Delta t} + \mathbf{b} + \mathbf{P} \lambda = \mathbf{0}, \quad (15)$$

$$\mathbf{P}^T \mathbf{u}^{t+\Delta t} = \mathbf{0}, \quad (16)$$

where  $\lambda = -\Delta t \mathbf{p}$ ,  $\mathbf{A} = \mathbf{M} + \Delta t \mathbf{D}$ , and  $\mathbf{b} = -\mathbf{M} \mathbf{u}^* + \Delta t \mathbf{B} \mathbf{f}$ . Solving this equation is equivalent to solving the quadratic optimization problem (Equation 2), since its first order optimality conditions are equivalent to Equations 15 and 16. We are interested in this perspective because it allows us to incorporate nonlinear terms into the model, in particular surface tension forces. Adding this term in the form of body forces yields lower accuracy and leads to a stringent stability time step restriction for surface-tension dominated flows. Instead, we add the surface energy term  $U(\mathbf{x})$  to our objective function

$$\frac{1}{2} \mathbf{u}^T \mathbf{A} \mathbf{u} + \mathbf{u}^T \mathbf{b} + U(\mathbf{x} + \Delta t \mathbf{u}). \quad (17)$$

We use a second-order Taylor series approximation for  $U(\mathbf{x} + \Delta t \mathbf{u})$

$$U(\mathbf{x} + \Delta t \mathbf{u}) \approx U(\mathbf{x}) + \Delta t \nabla U \mathbf{u} + \frac{1}{2} \Delta t^2 \mathbf{u}^T \nabla \nabla U \mathbf{u}, \quad (18)$$

which leads us to another quadratic optimization problem in the standard form with

$$\mathbf{A}' = \mathbf{A} + \Delta t^2 \nabla \nabla U, \quad (19)$$

$$\mathbf{b}' = \mathbf{b} + \Delta t \nabla U. \quad (20)$$

Surface energy is proportional to the free surface area  $A$  of the fluid  $U(\mathbf{x}) = \sigma A(\mathbf{x})$ . The constant of proportionality  $\sigma$  is called the *surface energy density* and it is a material constant with different values on contact surfaces between each pair of phases (liquid, gaseous and solid) in the system. In order to evaluate the gradient and the Hessian of the energy density ( $\nabla U$  and  $\nabla \nabla U$ ), we need to find the gradient and the Hessian of the area for each interface triangle. We can find symmetric formulas for those by applying a Taylor approximation to Heron's formula for the area  $A_t$  of a triangle  $t$  with vertices  $\mathbf{x}_i, \mathbf{x}_j, \mathbf{x}_k$ . Let us denote  $\mathbf{e}_\alpha = \mathbf{x}_\gamma - \mathbf{x}_\beta$ , where  $(\alpha, \beta, \gamma)$  is an even permutation of  $(i, j, k)$  and  $e_\alpha = \|\mathbf{e}_\alpha\|$ . Lengthy calculations lead to the following results

$$\nabla_{\alpha} A_t = \frac{(e_\alpha^2 - e_\beta^2 + e_\gamma^2) \mathbf{e}_\beta - (e_\alpha^2 + e_\beta^2 - e_\gamma^2) \mathbf{e}_\gamma}{8A_t}, \quad (21)$$

$$\nabla_{\alpha\alpha} A_t = \frac{2e_\alpha^2 \mathbf{I} - 2e_\alpha \mathbf{e}_\alpha^T - (\nabla_{\alpha} A_t)(\nabla_{\alpha} A_t)^T}{8A_t}, \quad (22)$$

$$\nabla_{\alpha\beta} A_t = \frac{(e_\gamma^2 - e_\alpha^2 - e_\beta^2) \mathbf{I} - \mathbf{e}_\gamma \mathbf{e}_\gamma^T + \mathbf{e}_\alpha \mathbf{e}_\alpha^T + \mathbf{e}_\beta \mathbf{e}_\beta^T - (\nabla_\alpha A_t)(\nabla_\beta A_t)^T}{8A_t}, \quad (23)$$

where  $\nabla_\alpha$  is the gradient operator with respect to the position of the vertex  $\mathbf{p}_\alpha$  and  $\nabla_{\alpha\beta} \equiv \nabla_\alpha \nabla_\beta$ . Note that Equation 21 is equivalent to the *cotangent formula* [PP93], commonly used in discrete exterior calculus. A comparison of the fluid simulation results using first-order and second-order surface energy approximations is presented in Section 4.1.

We have presented a finite element discretization of the Navier-Stokes equation, formulated in the form of a quadratic optimization problem, which fully couples the pressure and velocity fields and allows us to accurately include surface tension forces in our model.

### 3.4. Pressure Stabilization

In some cases, the matrix  $\mathbf{P}$  might not have full column rank, making the KKT matrix singular. In the finite element literature this is referred to as *locking*. To circumvent this problem we add a stabilization term to the KKT system (closely related to the concept of *regularization* in the field of machine learning and inverse problems).

We apply the idea of pseudo-compressibility [She97] to stabilize the Navier-Stokes equations. There are different versions of this class of pseudo-compressibility methods. However, the version we use replaces the continuity constraint  $\nabla \cdot \mathbf{u} = 0$  with

$$\nabla \cdot \mathbf{u} - \frac{\varepsilon}{\rho} \nabla^2 p = 0 \quad (24)$$

where  $\varepsilon$  is termed the *stabilization parameter* and is related to the time step one is using. Shen [She97] suggests using  $\varepsilon \approx \Delta t$ .

We can discretize this modified continuity equation using a finite volume method

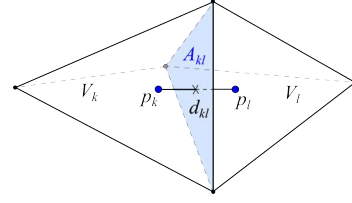
$$0 = \int_{\mathcal{V}_{\text{fluid}}} \left( \nabla \cdot \mathbf{u} - \frac{\Delta t}{\rho} \nabla^2 p \right) dV \approx \mathbf{P}^T \mathbf{u} - \Delta t \mathbf{S} p. \quad (25)$$

The formulation of the matrix  $\mathbf{P}$  has already been shown in Section 3.3. In order to evaluate the second term, we split the volume integral into the sum of integrals over each tetrahedron and apply Gauss' theorem, which yields

$$\frac{\Delta t}{\rho} \int_{\mathcal{V}_{\text{fluid}}} (\nabla^2 p) dV = \frac{\Delta t}{\rho} \sum_k \sum_l \int_{A_{kl}} (\nabla p \cdot \mathbf{n}_{kl}) dA, \quad (26)$$

where  $A_{kl}$  is a face of the  $k^{\text{th}}$  tetrahedron, shared with the  $l^{\text{th}}$  tetrahedron, and  $\mathbf{n}_{kl}$  is the normal vector to  $A_{kl}$ . Now, assuming that the pressure field is discretized at the barycenters of the elements in our mesh we can approximate the term  $\nabla p \cdot \mathbf{n}_{kl}$  on  $A_{kl}$ . We do it by evaluating the term  $p(\mathbf{x} + d\mathbf{n})$  using Taylor approximation

$$p(\mathbf{x} + d\mathbf{n}) \approx p(\mathbf{x}) + (\nabla p(\mathbf{x}) \cdot \mathbf{n}) d. \quad (27)$$



**Figure 4:** The distance  $d_k$  from the barycenter of a tetrahedron  $k$  to its face  $A_{kl}$  is four times smaller than its height  $h_k$  relative to that face. The volume of this tetrahedron  $V_k = \frac{1}{3} h_k A_{kl}$ , hence  $d_k = \frac{3V_k}{4A_{kl}}$ . Analogously,  $d_l = \frac{3V_l}{4A_{kl}}$ . From this, we have that the distance between the barycenter, in the direction orthogonal to  $A_{kl}$  equals  $d_{kl} = d_k + d_l = \frac{3}{4} \frac{V_k + V_l}{A_{kl}}$ .

From this

$$\nabla p \cdot \mathbf{n}_{kl} \approx \frac{p_l - p_k}{d_{kl}}, \quad (28)$$

where  $d_{kl}$  is the distance between the barycenters of the  $k^{\text{th}}$  and the  $l^{\text{th}}$  tetrahedra projected onto  $\mathbf{n}_{kl}$ . This is a good approximation as long as the barycenters of tetrahedra  $k$  and  $l$  project onto the same point on  $A_{kl}$ . Fortunately, the DSC method optimizes the mesh to favor this property. This approximation has been used previously by Chentenez et al. [CFL\*07], and similar approximations are also used in computational fluid dynamics [VM95].

We can express the formula shown in Equation 28 using the area of  $A_{kl}$  and the volumes of its adjacent tetrahedra

$$d_{kl} = \frac{3}{4} \frac{V_k + V_l}{A_{kl}} \quad (29)$$

(see Figure 4 for the explanation). Hence

$$\frac{\Delta t}{\rho} \int_{\mathcal{V}_{\text{fluid}}} (\nabla^2 p) dV \approx \frac{\Delta t}{\rho} \sum_k \sum_l \frac{A_{kl}}{d_{kl}} (p_l - p_k). \quad (30)$$

For the sake of brevity, let us denote

$$\delta_{kl} = \frac{A_{kl}}{d_{kl}} = \frac{4}{3} \frac{A_{kl}^2}{V_k + V_l}. \quad (31)$$

This way we can write the matrix  $\mathbf{S}$  as

$$\mathbf{S}_{kl} = \frac{1}{\rho} \begin{cases} \delta_{kl} & \text{if } k \neq l \\ -\sum_{m \neq k} \delta_{km} & \text{if } k = l \end{cases}, \quad (32)$$

where  $\delta_{kl}$  is given by Equation 31 if tetrahedra  $k$  and  $l$  share a face, or otherwise equals 0. Such a pressure stabilization term relaxes the incompressibility constraint by allowing limited volume exchange between adjacent tetrahedra, while keeping the total volume of the fluid constant.

We can easily include this term in our KKT system (Equation 4) by replacing the continuity constraint  $\mathbf{P}^T \mathbf{u} = \mathbf{0}$  with Equation 25, obtaining

$$\begin{bmatrix} \mathbf{A} & \mathbf{P} \\ \mathbf{P}^T & \mathbf{S} \end{bmatrix} \begin{bmatrix} \mathbf{u} \\ \lambda \end{bmatrix} = \begin{bmatrix} -\mathbf{b} \\ \mathbf{0} \end{bmatrix}. \quad (33)$$

The comparison of the fluid simulation results using this pressure stabilization scheme and the one by Misztal et al. [MBE\*10] is presented in Section 4.1.

### 3.5. Using a Preconditioned Iterative Solver

The KKT system solving step was the main bottleneck in the previous work [EMB11]. The fluid simulation method would spend up to 70% of the computation time solving the linear system using the Cholesky decomposition method. This is why we decided to use an iterative solver in our work. The indefiniteness of the modified KKT-matrix may cause numerical problems when we want to solve our system of linear equations. Ideally we would like to apply a scheme like the Conjugate Gradient (CG) method. However, CG typically does not converge well for indefinite systems.

Typically MINRES, SYMMLQ [PS75] or GMRES [SS86] are used instead of CG when dealing with an indefinite matrix. We use the Generalized Minimum Residual (GMRES) method. It is similar to CG except it keeps a memory limited local storage of vectors spanning the Krylov space that is being explored [Saa03]. One can find off-the-shelf GPU implementations of GMRES which can boost the performance with almost no programming effort. We use CUSP [BG10] as our GPU solver.

The matrix  $\mathbf{K}$  is not diagonally dominant, so we can not use the well-known Jacobi preconditioner. Instead, in order to improve the GMRES method's convergence rate, we apply a diagonal approximation of Murphy's block preconditioner [PTCL03]. It is based on the idea of using a diagonal version of the Schur complement as a preconditioner

$$\mathbf{P}_{\text{schur}} = \begin{bmatrix} \mathbf{A} & \mathbf{0} \\ \mathbf{0} & (\mathbf{S} - \mathbf{P}\mathbf{A}^{-1}\mathbf{P}^T) \end{bmatrix} \quad (34)$$

The diagonal approximation of this preconditioner would be

$$\mathbf{P}_{\text{diag}} = \begin{bmatrix} \text{diag}(\mathbf{A}) & \mathbf{0} \\ \mathbf{0} & \text{diag}(\mathbf{S} - \mathbf{P}(\text{diag}(\mathbf{A}))^{-1}\mathbf{P}^T) \end{bmatrix} \quad (35)$$

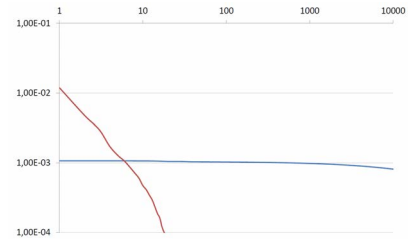
This preconditioner is inexpensive to compute. Furthermore,  $\mathbf{P}_{\text{diag}}$  is trivial to invert. Hence we solve the preconditioned system

$$\mathbf{P}_{\text{diag}}^{-1}\mathbf{K} \begin{bmatrix} \mathbf{u} \\ \lambda \end{bmatrix} = \mathbf{P}_{\text{diag}}^{-1} \begin{bmatrix} -\mathbf{b} \\ \mathbf{0} \end{bmatrix}. \quad (36)$$

In our experiments this seems to work well in combination with GMRES (see Figure 5 for convergence plots).

### 3.6. Solid Boundaries

In the computer graphics community there are two popular choices of boundary condition equations at the contact surface between the fluid and the solid boundaries. The *free-slip* condition states that at the solid boundaries the normal velocity of the fluid must be 0 (in case the solid wall is static) or



**Figure 5:** Typical convergence behavior for a GMRES solver using our preconditioner (red line) and without a preconditioner (blue line) in a flow dominated by surface tension: the horizontal axis shows the number of GMRES iterations and the vertical axis – the relative residual. In this example, preconditioning helps GMRES converge to a desired final residual of  $10^{-4}$  in as few as 20 iterations.

must match the normal velocity of the solid. This boundary condition is a popular choice for fluids with low viscosity values. The *no-slip* condition states that at the solid boundaries, the fluid does not move relative to the boundary (its velocity matches that of the solid). This boundary condition is favored when modeling fluids with high-viscosity values. In our experiments, we have been using the former approach, although implementation of a no-slip condition is also possible in our framework.

Let us focus on a static solid wall  $\mathcal{W} \subset \mathbf{R}^3$  (including moving solids is straight-forward and only changes the left-hand side part of our KKT system). Let us denote the set of all fluid vertices in contact with the solid boundary as

$$\mathcal{C} = \{k : \mathbf{p}_k \in \partial\mathcal{W}\}, \quad (37)$$

where  $\mathbf{p}_k$  is the position of the  $k^{\text{th}}$  vertex. We may now write the free-slip solid boundary condition for a vertex  $k \in \mathcal{C}$  as

$$\mathbf{n}_k^T \mathbf{u}_k = 0 \quad \forall k \in \mathcal{C}, \quad (38)$$

where  $\mathbf{u}_k$  is the fluid's velocity at the  $k^{\text{th}}$  vertex and  $\mathbf{n}_k$  is the normal to the boundary at  $\mathbf{p}_k$ . Given the velocity field  $\mathbf{u} \in \mathbf{R}^{3N_V}$  we may now define the boundary condition at solid walls as

$$\mathbf{C}\mathbf{u} = \mathbf{0} \quad (39)$$

where  $\mathbf{C} \in \mathbf{R}^{|\mathcal{C}| \times 3N_V}$ . Now we may add the solid boundary conditions to our optimization problem as a hard constraint

$$\mathbf{u}^{t+\Delta t} = \arg \min_{\mathbf{u}} \frac{1}{2} \mathbf{u}^T \mathbf{A} \mathbf{u} + \mathbf{u}^T \mathbf{b} \quad (40)$$

subject to

$$\mathbf{P}^T \mathbf{u} = \mathbf{0} \quad (41)$$

$$\mathbf{C}\mathbf{u} = \mathbf{0} \quad (42)$$

This results in a KKT-matrix

$$\mathbf{K}' = \begin{bmatrix} \mathbf{A} & \mathbf{P} & \mathbf{C}^T \\ \mathbf{P}^T & \varepsilon \mathbf{S} & \mathbf{0} \\ \mathbf{C} & \mathbf{0} & \mathbf{0} \end{bmatrix} \quad (43)$$

This is clearly a symmetric indefinite matrix. The  $\mathbf{C}$ -matrix has full row rank and therefore the KKT-matrix  $\mathbf{K}'$  is nonsingular. For such a KKT-matrix, we modify our preconditioner as follows

$$\mathbf{P}'_{\text{diag}} = \begin{bmatrix} \mathbf{P}^{\text{diag}} & \mathbf{0} \\ \mathbf{0} & \text{diag}(-\mathbf{C}^T (\text{diag}(\mathbf{A}))^{-1} \mathbf{C}) \end{bmatrix} \quad (44)$$

### 3.7. Multiple phases

One can easily adapt the DSC method so that it handles multiple phases. Instead of having just two labels for tetrahedra (*inside* and *outside*), one can use an arbitrary number of labels, each representing a different phase. This allows us to simulate several, immiscible fluids with different density, surface tension and viscosity values.

Since each tetrahedron in the mesh is occupied by just one fluid, the solver remains essentially unchanged. We apply full-slip boundary conditions on the contact surface between each pair of interacting fluids, meaning that the vertices on the interface between two fluids are given freedom to move in every direction. The discretization of the Navier-Stokes equation presented in Section 3.3 remains valid when we associate different density, viscosity and surface energy density values with different elements. The only part that needs changing is the pressure stabilization term. In order to avoid exchanging volume between two different fluids, we modify the matrix  $\mathbf{S}$  (given by Equation 32) as follows

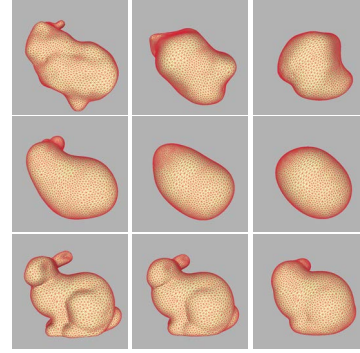
$$\mathbf{S}_{kl} = \frac{1}{\rho_i} \begin{cases} \delta_{kl} & \text{if } k \neq l \\ -\sum_{m \neq k} \delta_{km} & \text{if } k = l \end{cases}, \quad (45)$$

where  $\delta_{kl}$  is given by Equation 31 if tetrahedra  $k$  and  $l$  share a face and belong to the same fluid (have the same label  $i$ ), or otherwise equals 0.

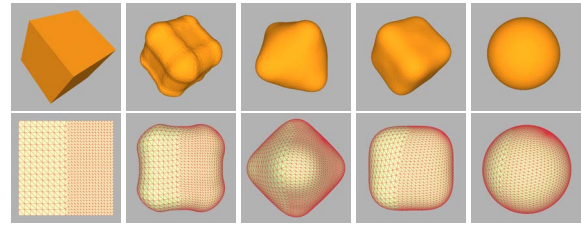
## 4. Tests and Results

### 4.1. Viscosity

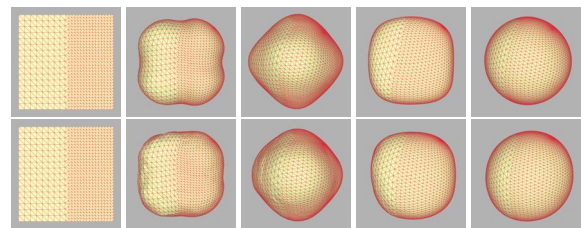
In order to validate our viscosity model, we have run a simple experiment, in which a Stanford bunny model, given different viscosity coefficient values, deforms freely due to the surface tension force (Figure 6). The results of the experiment follow the intuition: when the viscosity coefficient is low, the fluid volume deforms rapidly, however it takes a long time to lose its kinetic energy and keeps oscillating; as we increase the viscosity coefficient, we introduce more damping — the deformation progresses more slowly and the initial shape smoothly transitions into an oval, and further on — into a sphere.



**Figure 6:** Stanford bunny model deforming in zero gravity due to the surface tension forces after (from the left to the right) 1, 2 and 3 seconds. The fluid's viscosity, from the top to the bottom:  $\mu = 0 P$  (the unit of viscosity),  $\mu = 0.1 P$  and  $\mu = 1 P$ .



**Figure 7:** A uniformly (top row) and non-uniformly (bottom row) tessellated cube in zero gravity deforming due to surface tension forces. Rather than deforming directly into a sphere, the blob of fluid oscillates rapidly between an octahedron-like and a cube-like shape until its kinetic energy dissipates and it becomes spherical. Notice that that non-uniform tessellation of the initial volume of fluid does not affect the behavior of the fluid significantly, nor does it introduce ghost forces.



**Figure 8:** Capillary waves experiment results with first-order surface energy term (top row) and with the previous pressure stabilization scheme from [MBE\* 10] (bottom row). In the former case, the behavior of the fluid is similar to that presented in Figure 7, however, noticeable asymmetry and slight drift emerge. In the latter case, simulation quality is significantly lower and the capillary waves are not captured correctly.

## 4.2. Capillary waves

One of the applications requiring both low numerical diffusion and accurate treatment of the surface tension forces is the simulation of capillary waves. The discussion of the problem and benchmark results have been provided by Brochu et al. [BBB10]. We have repeated one of their experiments to see how our method deals with this problem. Our results are in agreement with previous work (they are presented in Figure 7). Note that the simulation results do not depend significantly on the initial tessellation of the fluid volume. This is the case, however, in the earlier approaches by Misztal et al. [MBE\*10] and Erleben et al. [EMB11] (as shown in Figure 8). While using the first-order surface energy approximation leads to generally sane results, the free surface of the fluid quickly becomes visibly asymmetric, and the fluid volume begins to drift. Using the pressure stabilization scheme from [MBE\*10] dramatically deteriorates the simulation quality, introduces ghost forces (causing the drift of the fluid) and practically prevents us from capturing the capillary effects at all.

## 4.3. Other experiments

In order to verify our model, we have performed a “crown” experiment in which a spherical droplet falls into a shallow layer of fluid. The results of the simulations are shown in Figure 9 and they follow the intuition. Observe that proper handling of thin sheets of fluid comes naturally in our method.

Figures 1 and 10 present the results of our experiments with multiple immiscible fluids: in particular water and oil. Each type of contact between fluid, solid and gaseous phases is assigned different surface energy densities. We observe qualitatively different behavior in the different phases.

## 4.4. Performance

The statistics of our simulations are presented in Table 1. The timings are comparable to other finite element based simulation methods [WRK\*10]. By applying an iterative solver, we have significantly decreased the time spent on solving the KKT system. The DSC method’s mesh improvement functionality seems to work robustly, particularly for single phase simulations, where it allows us to keep most of the dihedral angles in the range  $10^\circ - 160^\circ$  except for the times when collisions occur. Those times, unfortunately, tend to introduce low quality tetrahedra which might not be removed immediately. While the fluid simulation method seems to deal with such elements rather well, they negatively affect the performance during the advection step. We are planning to address this issue by investigating more sophisticated mesh refinement schemes.

We have run all our experiments on a machine with an Intel® Core™ i7 CPU X 980 3.33GHz with an NVIDIA® Geforce™ GTX580 GPU.

## 5. Summary and Discussion

The distinguishing characteristic of our scheme is that it is Lagrangian with an explicit interface representation, yet also volumetric, using a single irregular grid for both simulation as well as tracking and handling collisions of parts of the interface. In this work, we have demonstrated that the method can deal with multiphase flows and that the qualitative behavior of the simulated fluid is as expected.

Our new pressure stabilization strategy resulted in lower numerical diffusion than in [MBE\*10, EMB11], allowing us to capture the capillary waves correctly and making our simulations of surface tension dominated flows on par with the state-of-the-art methods [BBB10, TWGT10]. Thin sheets are handled accurately without the need for any special treatment. Furthermore, our new pressure stabilization scheme made the method insensitive to the mesh element size, removing the problem of ghost forces present in earlier works when the initial tessellation of the fluid volume is non-uniform. This way, we have opened the doors for an adaptive-resolution, multi-scale fluid simulation using our framework.

Compared to [MBE\*10], we have also improved the treatment of solid boundaries. The presented formulation works well with the iterative linear system solver and simplifies adding moving solids to the model in the future, in contrast to the approach presented in [MBE\*10]. The use of a preconditioned iterative solver allows us to decrease the amount of time spent on solving the linear system, which was the bottleneck in [EMB11].

In the future we would like to further improve the performance of our fluid simulation method. One way of doing that would be by using a Krylov solver which takes symmetry into account (unlike GMRES) which would help obtain better quality results in a shorter time. We are also planning to investigate different mesh refinement schemes, which would allow us to improve the computational mesh quality when changes in the surface mesh topology take place. We would also like to explore the applicability of our method in simulating interactions between fluids and deformable solid bodies.

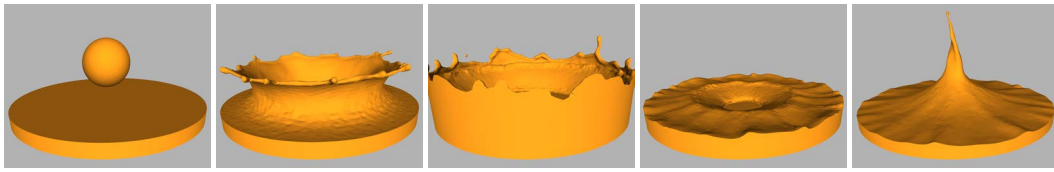
## Acknowledgements

This work was funded by a grant from the Danish Agency for Science, Technology and Innovation. We would like to thank anonymous reviewers for feedback.

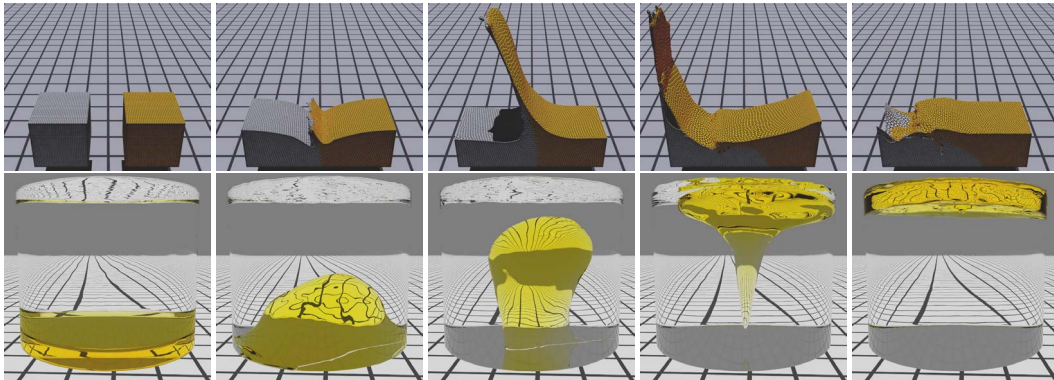
## References

- [BBB10] BROCHU T., BATTY C., BRIDSON R.: Matching fluid simulation elements to surface geometry and topology. In *ACM SIGGRAPH 2010 papers* (2010), ACM, p. XX. 2, 8
- [BG10] BELL N., GARLAND M.: Cusp: Generic parallel algorithms for sparse matrix and graph computations, 2010. Version 0.1.0. 6





**Figure 9:** A spherical droplet splashing in a cylindrical container with a shallow layer of water, producing a “crown”. Observe that our method does not have any problems with handling thin layers of fluid.



**Figure 10:** The results of our two-phase experiments with water and oil: the “double dam breaking” experiment in which the collision of the volumes of oil and water produces a rapidly moving jet (top); the movement of a small volume of oil in water due to the difference in densities (bottom).

- [BMF07] BRIDSON R., MÜLLER-FISCHER M.: Fluid simulation: Siggraph 2007 course notes. In *SIGGRAPH '07: ACM SIGGRAPH 2007 courses* (New York, NY, USA, 2007), ACM, pp. 1–81. 2
- [Bri08] BRIDSON R.: *Fluid Simulation*. A. K. Peters, Ltd., Natick, MA, USA, 2008. 2
- [BWHT07] BARGTEIL A. W., WOJTAN C., HODGINS J. K., TURK G.: A finite element method for animating large viscoplastic flow. *ACM Trans. Graph.* 26 (July 2007). 2
- [CFL\*07] CHENTANEZ N., FELDMAN B. E., LABELLE F., O'BRIEN J. F., SHEWCHUK J. R.: Liquid simulation on lattice-based tetrahedral meshes. In *SCA '07: Proc. of the 2007 ACM SIGGRAPH/Eurographics Symp. on Comp. animation* (2007), pp. 219–228. 2, 5
- [CGFO06] CHENTANEZ N., GOKTEKIN T. G., FELDMAN B. E., O'BRIEN J. F.: Simultaneous coupling of fluids and deformable bodies. In *SCA '06: Proc. of the 2006 ACM SIGGRAPH/Eurographics Symp. on Comp. animation* (2006), pp. 83–89. 2
- [CMD\*00] CHANDRA R., MENON R., DAGUM L., KOHR D., MAYDAN D., McDONALD J.: *Parallel Programming in OpenMP*. Morgan Kaufmann, 2000.
- [EMB11] ERLEBEN K., MISZTAL M. K., BÆRENTZEN J. A.: Mathematical foundation of the optimization-based fluid animation method. In *Proc. of the 2011 ACM SIGGRAPH/Eurographics Symp. on Comp. Animation* (2011), pp. 101–110. 2, 4, 6, 8
- [ETK\*07] ELCOTT S., TONG Y., KANSO E., SCHRÖDER P., DESBRUN M.: Stable, circulation-preserving, simplicial fluids. *ACM Trans. Graph.* 26, 1 (2007), 4. 2
- [FF01] FOSTER N., FEDKIW R.: Practical animation of liquids. In *SIGGRAPH '01: Proc. of the 28th annual conference on Comp. graphics and interactive techniques* (2001), ACM, pp. 23–30. 2
- [FM96] FOSTER N., METAXAS D.: Realistic animation of liquids. *Graph. Models Image Process.* 58, 5 (1996), 471–483. 2
- [FM97] FOSTER N., METAXAS D.: Modeling the motion of a hot, turbulent gas. In *SIGGRAPH '97: Proc. of the 24th annual conference on Comp. graphics and interactive techniques* (1997), pp. 181–188. 2
- [FOG97] FREITAG L. A., OLLIVIER-GOOCH C.: Tetrahedral mesh improvement using swapping and smoothing. *International Journal for Numerical Methods in Engineering* 40 (1997), 3979–4002. 3
- [FOK05] FELDMAN B. E., O'BRIEN J. F., KLINGNER B. M.: Animating gases with hybrid meshes. In *SIGGRAPH '05: ACM SIGGRAPH 2005 Papers* (New York, NY, USA, 2005), ACM, pp. 904–909. 2
- [FOK05] FELDMAN B. E., O'BRIEN J. F., KLINGNER B. M., GOKTEKIN T. G.: Fluids in deforming meshes. In *SCA '05: Proc. of the 2005 ACM SIGGRAPH/Eurographics Symp. on Comp. animation* (2005), ACM, pp. 255–259. 2
- [FSJ01] FEDKIW R., STAM J., JENSEN H. W.: Visual simulation of smoke. In *SIGGRAPH '01: Proc. of the 28th annual conference on Comp. graphics and interactive techniques* (New York, NY, USA, 2001), ACM, pp. 15–22. 2
- [Jia07] JIAO X.: Face offsetting: A unified approach for explicit moving interfaces. *J. Comput. Phys.* 220, 2 (2007), 612–625. 3
- [KFCO06] KLINGNER B. M., FELDMAN B. E., CHENTANEZ

example	#tets	#triangles	average time per iteration (seconds)			
	initial/max/median	initial/max/median	assembly	GMRES	advection	total
CUBE N.U.	20345 / 20345 / 17403	7616 / 7616 / 7610	6.41	3.48	3.87	13.9
OIL/WATER	42319 / 42319 / 24538	8832 / 9532 / 9040	8.91	6.57	4.19	19.9
TEASER	24770 / 92893 / 43983	11840 / 37089 / 20575	22.0	11.2	14.4	48.1
CROWN	33810 / 94789 / 81983	15104 / 38054 / 32160	32.7	19.6	19.8	72.8
DAM BREAKING	84852 / 108518 / 98722	33370 / 42736 / 38291	40.2	21.4	23.1	85.6

**Table 1:** Simulation statistics: the initial number of tetrahedra of the embedding mesh (excluding tetrahedra labeled outside which are not used for computation); the initial/maximum/median number of tetrahedra of the computational mesh; the initial/maximum/median number of the surface elements; the average timings of each step of the simulation: matrix assembly, solving the linear system using the GMRES method, advection step using the DSC method (including mesh improvement); the average timing of a single iteration.

- N., O'BRIEN J. F.: Fluid animation with dynamic meshes. *ACM Trans. Graph.* 25, 3 (2006), 820–825. 2
- [Kim10] KIM B.: Multi-phase fluid simulations using regional level sets. *ACM Trans. Graph.* 29 (December 2010), 175:1–175:8. 2
- [KS07] KLINGNER B. M., SHEWCHUK J. R.: Agressive tetrahedral mesh improvement. In *Proc. of the 16th International Meshing Roundtable* (Oct. 2007), pp. 3–23. 3
- [LSSF06] LOSASSO F., SHINAR T., SELLE A., FEDKIW R.: Multiple interacting liquids. *ACM Trans. Graph.* 25, 3 (2006), 812–819. 2
- [MB12] MISZTAL M. K., BÆRENTZEN J. A.: Topology adaptive interface tracking using the deformable simplicial complex. *ACM Trans. Graph.* 31, 3 (June 2012), 24:1–24:12. 1, 3
- [MBE\*10] MISZTAL M. K., BRIDSON R., ERLEBEN K., BÆRENTZEN J. A., ANTON F.: Optimization-based fluid simulation on unstructured meshes. In *Proc. of the 7th Workshop on Virtual Reality Interactions and Physical Simulations, VRIPHYS* (2010), pp. 11–20. 2, 6, 7, 8
- [Mis10] MISZTAL M. K.: *Deformable Simplicial Complexes*. PhD thesis, Technical University of Denmark (DTU), Denmark, 2010. 1, 3
- [Mon92] MONAGHAN J. J.: Smoothed particle hydrodynamics. *Annual review of astronomy and astrophysics* 30 (1992), 543–574. 2
- [MST10] MCADAMS A., SIFAKIS E., TERAN J.: A parallel multigrid poisson solver for fluids simulation on large grids. In *Proc. of the 2010 ACM SIGGRAPH/Eurographics Symp. on Comp. Animation* (2010), pp. 65–74. 2
- [NW99] NOCEDAL J., WRIGHT S. J.: *Numerical optimization*. Springer Series in Operations Research. Springer-Verlag, New York, 1999. 4
- [PP93] PINKALL U., POLTHIER K.: Computing discrete minimal surfaces and their conjugates. *Experimental mathematics* 2, 1 (1993), 15–36. 5
- [PS75] PAIGE C. C., SAUNDERS M. A.: Solution of sparse indefinite systems of linear equations. *SIAM Journal on Numerical Analysis* 12, 4 (1975), 617–629. 6
- [PTCL03] PHOON K. K., TOH K. C., CHAN S. H., LEE F. H.: *Computational Fluid and Solid Mechanics 2003: Second Mit Conference 2003*, vol. 1. Elsevier Science Ltd, 2003, ch. A
- Generalised Jacobi Preconditioner for Finite Element Solution of Large-Scale Consolidation Problems, pp. 573–577. 6
- [Saa03] SAAD Y.: *Iterative methods for sparse linear systems*. Society for Industrial Mathematics, 2003. 6
- [She97] SHEN J.: Pseudo-compressibility methods for the unsteady incompressible navier-stokes equations. In *Proc. of the 1994 Beijing Symp. on Nonlinear Evolution Equations and Infinite Dynamical Systems* (1997), ZhongShan University Press, pp. 68–78. 5
- [SS86] SAAD Y., SCHULTZ M. H.: GMRES: A generalized minimal residual method for solving nonsymmetric linear systems. 6
- [SSP07] SOLENTHALER B., SCHLÄFLI J., PAJAROLA R.: Pajarola r.: A unified particle model for fluid solid interactions: Research articles. *Comput. Animat. Virtual Worlds* 18 (2007), 69–82. 2
- [Sta99] STAM J.: Stable fluids. In *SIGGRAPH '99: Proceedings of the 26th annual conference on Comp. graphics and interactive techniques* (New York, NY, USA, 1999), ACM Press/Addison-Wesley Publishing Co., pp. 121–128. 2
- [TWGT10] THÜREY N., WOJTAN C., GROSS M., TURK G.: A multiscale approach to mesh-based surface tension flows. In *SIGGRAPH '10: ACM SIGGRAPH 2010 papers* (New York, NY, USA, 2010), ACM, pp. 1–10. 8
- [VM95] VERSTEEG H. K., MALALASEKERA W.: *An introduction to computational fluid dynamics: the Finite Volume method*. Longman Scientific and Technical, 1995. 5
- [WBOL07] WENDT J. D., BAXTER W., OGUZ I., LIN M. C.: Finite volume flow simulations on arbitrary domains. *Graph. Models* 69, 1 (2007), 19–32. 2
- [WRK\*10] WICKE M., RITCHIE D., KLINGNER B. M., BURKE S., SHEWCHUK J. R., O'BRIEN J. F.: Dynamic local remeshing for elastoplastic simulation. *ACM Trans. Graph.* 29 (July 2010), 49:1–49:11. 2, 8
- [WT08] WOJTAN C., TURK G.: Fast viscoelastic behavior with thin features. In *SIGGRAPH '08: ACM SIGGRAPH 2008 papers* (New York, NY, USA, 2008), ACM, pp. 1–8. 2
- [ZYP06] ZHENG W., YONG J.-H., PAUL J.-C.: Simulation of bubbles. In *Proc. of the 2006 ACM SIGGRAPH/Eurographics Symp. on Comp. animation* (2006), SCA '06, Eurographics Association, pp. 325–333. 2