Interaction of Fluids with Deformable Solids

Matthias Müller Simon Schirm Matthias Teschner Bruno Heidelberger Markus Gross
ETH Zürich, Switzerland

Abstract

In this paper, we present a method for simulating the interaction of fluids with deformable solids. The method is designed for the use in interactive systems such as virtual surgery simulators where the real-time interplay of liquids and surrounding tissue is important.

In computer graphics, a variety of techniques have been proposed to model liquids and deformable objects at interactive rates. As important as the plausible animation of these substances is the fast and stable modeling of their interaction. The method we describe in this paper models the exchange of momentum between Lagrangian particle-based fluid models and solids represented by polygonal meshes. To model the solid-fluid interaction we use virtual boundary particles. They are placed on the surface of the solid objects according to Gaussian quadrature rules allowing the computation of smooth interaction potentials that yield stable simulations. We demonstrate our approach in an interactive simulation environment for fluids and deformable solids.

Keywords: smoothed particle hydrodynamics (SPH), finite element method (FEM), fluid-solid interaction

1 Introduction

Interactive physically based simulation is a rapidly growing research area with an increasing number of applications, e. g. in games and computational surgery. In these simulation environments, deformable objects play an important role. For the simulation of deformable solids, a variety of models have been proposed ranging from efficient mass-spring approaches to methods based on the physically more accurate Finite Element Method (FEM). Some of these methods allow the simulation of elastically and plastically deformable solids at interactive speed.

More recently, there has been an increased interest in efficient methods for the realistic simulation of fluids. These approaches can be employed to represent blood or other liquids. Besides deformable models, they play an essential role in applications such as surgery simulation. So far, only a few interactive methods for the simulation of fluids with free surfaces have been proposed.

With the ability to simulate both, deformable solids and fluids, a new problem has been introduced, namely the modeling of the interaction of these structures. An interaction model suitable for the use in interactive environments needs to be computationally efficient and the generated interaction forces must not induce any instabilities to the dynamic simulation.

In this paper, we present a new technique to model interactions between particle based fluids and mesh based deformable solids which meets these constraints. We present our interaction model with fluids represented by a Smoothed Particle Hydrodynamics approach (SPH) and with deformable solids represented by a Finite Element approach. However, the general interaction model we propose works with any type of deformation technique as long as the object surface is represented by a polygonal mesh and the fluid by Lagrangian particles.

2 Related Work

The majority of publications in the area of physically based animation focuses on physical systems of one single type. Deformable objects are interesting to study in their own right. In fluid simulation, on the other hand, boundary conditions are often considered a necessary but not a central issue. They are typically derived from simple geometric primitives. Our method connects these two areas of research.

In the field of computer graphics, a large number of mesh based methods for the physically based simulation of deformable objects have been proposed since the pioneering paper of Terzopoulos [1]. Early techniques were mostly based on mass-spring systems, which are still popular for cloth simulation [2, 3]. More recent methods discretize continuous elasticity equations via the Boundary Element Method (BEM) [4] or the Finite Element Method (FEM) [5, 6, 7].

Since T. Reeves [8] introduced particle systems as a technique for modeling fuzzy objects twenty years ago, a variety of special purpose, partice based fluid simulation techniques have been developed in the field of computer graphics. Desbrun and Cani [9] where among the first to use Smoothed Particle Hydrodynamics (SPH) [10] to derive interaction forces for particle systems. They added space-adaptivity in [11]. Later, Stora et al. [12] used a similar particle based model to animate lava flows. In [13], Müller et al. derived inter particle forces from SPH and the Navier Stokes equation to simulate water with free surfaces at interactive rates. Recently, Premoze et al. [14] introduced the Moving-Particle Semi-Implicit (MPS) method to computer graphics for the simulation of fluids. As a mesh-free method, it is closely related to SPH but in contrast to standard SPH, it allows the simulation

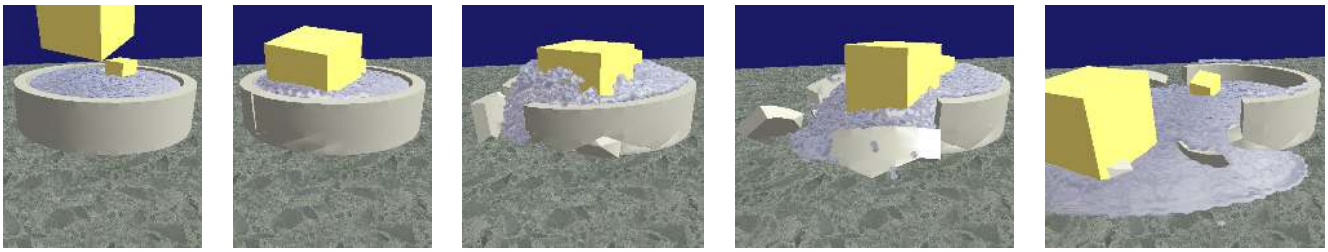


Figure 1: A box falls into a pool and generates a shock wave that causes the pool to fracture.

of incompressible fluids. In all these papers, boundary conditions are not treated explicitly. The fluids typically interact with solid walls or the ground.

Genevaux et al. [15] address the interaction problem explicitly. They propose a method to simulate the interaction between solids represented by mass-spring networks and an Eulerian fluid grid by applying spring forces to the mass-less marker particles in the fluid and the nodes of the mass-spring network. However, solids are typically represented by coarse meshes, especially in interactive simulations. Thus, the nodes of a mass-spring network are not very well suited for the application of interaction forces. Therefore, Monaghan, one of the founders of the SPH formalism, uses special boundary or ghost particles on fixed borders to model interactions [16]. The idea of ghost particles was picked up in several following projects including our own. The key contribution of our paper is to place these ghost particles onto boundary triangles of deformable objects and to derive their locations and weights according to Gauss integration [17], which allows to model fluid-solid interactions stably at interactive rates.

3 Physical Problem Description

In physically based animation, we are interested in the simulation of macroscopic effects at interactive speed. Therefore, we consider macroscopic models for both, solids and fluids. Materials, which are homogeneous at the macroscopic level, can mathematically be described as a continuum [18]. Thereby, quantities such as the density ρ, viscosity μ, deformation $\mathbf{u}$ or velocity $\mathbf{v}$ are all mathematically expressed by continuous functions over space and time. A physical model relates these quantities to each other via partial differential equations (PDEs). The mechanical behavior of an elastic solid can be described by the following equation

$$\rho \frac{\partial^2}{\partial t^2}\mathbf{u} = \nabla \cdot \sigma_s(\mathbf{u}) + \mathbf{f}, \tag{1}$$

which expresses Newton's equation of motion, namely that the change of momentum on the left hand side is equal to the internal elastic forces due to the stresses σ_s plus the externally applied body forces $\mathbf{f}$. The stresses σ_s are functions of the displacements $\mathbf{u}$. The equation is in Lagrangian form since the displacement vectors $\mathbf{u}$ follow the material points.

Similarly, mechanical properties of incompressible Newtonian fluids can be described by the following two equations in Eulerian form where fluid quantities are observed in a fixed coordinate frame

$$\rho \left(\frac{\partial \mathbf{v}}{\partial t} + \mathbf{v} \cdot \nabla \mathbf{v} \right) = \nabla \cdot \sigma_f(\mathbf{v}) + \mathbf{f} \tag{2}$$

$$\nabla \cdot \mathbf{v} = 0. \tag{3}$$

Equation (2) again states that the change of momentum equals the internal forces derived from the stresses σ_f plus the externally applied body forces $\mathbf{f}$. The stress tensor $\sigma_f = 2\mu\epsilon(\mathbf{v}) - p\mathbf{I}$ is composed of the viscosity stress and the pressure stress. The viscosity stress is dependent on the viscosity μ and the strain rate tensor ϵ while the pressure stress only depends on the scalar pressure p. For an incompressible fluid, the velocity field is divergence free (Eq. (3)).

Comparison of the right hand side of the two equations of motion (1) and (2) reveals, that the Eulerian description makes the additional convection term $\mathbf{v} \cdot \nabla \mathbf{v}$ necessary. In Sec. 4, we discuss numerical methods to solve both equations of motion. For fluids we focus on particle based methods such as SPH for which this term can be omitted.

3.1 Boundary Conditions

Materials such as fluids or solids are bounded by spatial limits. The behavior of materials at these limits is defined by boundary conditions. The boundary conditions relate the quantities of the two adjacent materials to each other at the interface. In the case of fluid-solid interaction, the geometrical domain of the interface Γ is defined as a surface between the volumetric solid continuum and the volumetric fluid continuum (see Fig. 2(a)). We focus on three main types of boundary conditions.

3.2 No-Penetration Condition

If the solid is considered to be impermeable, no fluid element is allowed to cross the boundary, which is described in the following equation:

$$(\frac{\partial}{\partial t}\mathbf{u} - \mathbf{v}) \cdot \mathbf{n} = 0 \quad \text{at the boundary } \Gamma, \tag{4}$$

where $\mathbf{n}$ is the normal on Γ (see Fig. 2(b)). The equation states that the components of the velocities of the fluid and the deformable object perpendicular to Γ are equal.

3.3 No-Slip Condition

The no-slip condition models friction between the fluid and the solid (see Fig. 2(c)). It holds for most fluids-solid surfaces and it states that the velocity components tangential to the fluid surface have to be equal, thus

$$(\frac{\partial}{\partial t}\mathbf{u} - \mathbf{v}) \times \mathbf{n} = \mathbf{0} \quad \text{at the boundary } \Gamma. \tag{5}$$

If both independent boundary conditions (4) and (5) hold, we simply have $\frac{\partial}{\partial t}\mathbf{u} = \mathbf{v}$ at the boundary, i.e. both materials have the same velocity at the boundary.

3.4 Actio = Reactio

Newton's Third Law demands the continuity of stresses σ_s and σ_f throughout the boundary (see Fig. 2(d)). In other words, the traction forces of the solid $\mathbf{g}_f$ must be opposite to the traction forces of the fluid $\mathbf{g}_s$ on the boundary Γ

$$\mathbf{g}_s = \sigma_s \mathbf{n} = \sigma_f(-\mathbf{n}) = -\mathbf{g}_f, \tag{6}$$

where $\mathbf{n}$ is the outward normal on the solid and $-\mathbf{n}$ the outward normal on the fluid.

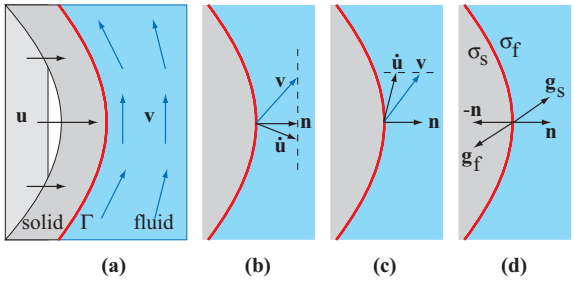


Figure 2: Boundary conditions: (a) A solid object is deformed by a displacement field $\mathbf{u}$ and interacts with a fluid whose velocity field is $\mathbf{v}$. The no-penetration condition is shown in image (b) where $\dot{\mathbf{u}}$ is the time derivative of $\mathbf{u}$. Image (c) illustrates the no-slip condition and image (d) Newton's Third Law.

4 Computational Model

The continuous equations and boundary conditions described in the previous section need to be discretized in space and time via a numerical method before they can be used in a computer simulation. We do not go into the details of how equation (1) for elastic objects can be solved numerically. For possible solutions using the Finite Element Method (FEM) we refer the reader to [19], [6] or [7]. All we require for our interaction method to work is

- that the solid object is represented by a mesh and
- that the displacements, velocities and forces are carried by the nodes of the mesh.

Most of the methods used in computer graphics to simulate deformable objects meet these constraints including mass-spring systems, the Finite Volume Method (FVM) and the Boundary Element Method (BEM).

For the simulation of fluids, two main numerical methods have been used in the field of physically based animation so far, namely Eulerian grid-based approaches [20, 21, 22] and Lagrangian methods based on particles (see Sec. 2). In this paper we concentrate on Lagrangian methods because they allow fluids with free surfaces to move freely in space while in the Eulerian case fluid computations are restricted to a spatially fixed and bounded grid. From the fluid simulation method we require

- that the fluid is represented by a set of particles and
- that positions, velocities and internal forces are carried by the particles.

Interaction modeling, thus, reduces to the problem of simulating the interaction between particles and triangulated surfaces.

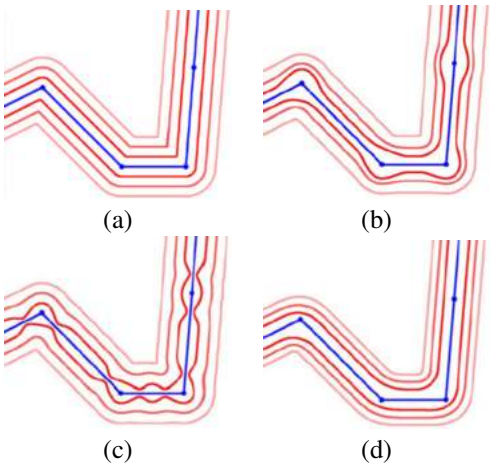


Figure 3: (a) Iso surfaces of the Euclidean distance field of a piecewise linear curve (blue) with discontinuous first derivatives near concavities. (b) Weighted sums yield smooth iso surfaces with bulges. (c) Normalization does not remove the artifact. (d) Convolution yields bulge-free smooth iso surfaces.

4.1 Interaction of Particles with Triangles

In physics, interaction potentials of two objects always depend on the distance between them. While the Euclidean distance between two points is uniquely defined, the distance between a point and a triangle or a point and a triangulated surface needs to be defined. Let us define the distance of a point $\mathbf{p}$ from a triangle t as

$$d(\mathbf{p}, t) = \min_{\mathbf{x} \in t} |\mathbf{p} - \mathbf{x}|, \tag{7}$$

and the distance of a point $\mathbf{p}$ from a triangulated surface T as

$$d(\mathbf{p}, T) = \min_{t \in T} d(\mathbf{p}, t). \tag{8}$$

Figure 3(a) shows several isosurfaces of the resulting distance field which is C^0 continuous everywhere. Unfortunately, concavities as well as close disconnected meshes generate discontinuous first derivatives of the distance field. Those discontinuities lead to discontinuous derivatives in forces since the forces depend on the distance field. A force field with discontinuous first derivatives, in turn, yields artifacts such as the so called *cooking* of particles in concave regions and reduced stability of the simulation.

The source of the discontinuity in the first derivatives is the minimum operator in Eqn. (8). One way to remove the problem is to replace the minimum by a weighted sum. Let the kernel $W(d, h) \in C^1$ be a positive smooth monotonously decreasing function which is zero for $d \geq h$ and has a vanishing derivative at $d = h$. We can then define the potential Φ of a point $\mathbf{p}$ with respect to a triangulated surface T which is not a Eucledian distance anymore

$$\Phi(\mathbf{p}, T) = \sum_{t \in T} d(\mathbf{p}, t) W(d(\mathbf{p}, t), h), \tag{9}$$

but which is C^0 and C^1 continuous everywhere. However, as Fig. 3(b) shows, the resulting field is distorted near triangle boundaries. This effect can be removed by normalization

$$\bar{\Phi}(\mathbf{p}, T) = \begin{cases} \frac{1}{w} \Phi(\mathbf{p}, T) & \text{if } w > 0 \\ 0 & \text{otherwise,} \end{cases} \tag{10}$$

where $w = \sum_{t \in T} W(d(\mathbf{p}, t), h)$. Unfortunately, normalization just distributes the distortions to adjacent regions of triangle interfaces as Fig. 3(c) shows. Another difficulty introduced by the weighted field method is the choice of the support radius h with respect to the size of the features of the boundary T. For large supports, small features are smoothed out while small supports reduce the interaction range of T.

4.2 Convolution Surfaces

The problems mentioned in the previous section are well known in the field of implicit surface modeling introduced by Blinn [23]. An elegant way to generate a bulge-free surface around a skeleton S, is to define a scalar function F_S as the convolution

$$F_S(\mathbf{p}) = \int_{\mathbf{x} \in S} W(\mathbf{p} - \mathbf{x}) d\mathbf{x}. \tag{11}$$

The implicit surface is defined by selecting an iso-surface of F_S. By replacing the skeleton S with the triangulated surface T we get a smooth potential field around T (see Fig. 3(d)). The problem with the weighted sum approach arises when when multiple triangles meet. In this case, all triangles contribute as a whole to the sum and generate bulges. In contrast, the convolution integral sums up infinitesimal parts of the skeleton each properly weighted (see Fig. 4). When the convolution integral is used, the interaction of $\mathbf{p}$ with the surface T is modeled as the interaction of $\mathbf{p}$ with all the infinitesimal points in T. For skeletal elements other than points, the integral in Eqn. (11) yields complex computations. Approaches to approximate this integral were proposed by Bloomenthal [24] and Sherstyuk [25]. Bloomenthal uses radial Gauss kernels which can be separated with respect to different dimensions. The separation allows post evaluation of the convolution in 3D space, only considering the distance to the triangle plane. Sherstyuk discovered a special kernel which can be analytically convoluted over a triangle domain. Neither method is suitable for computing physical interactions because we are not free in the choice of the kernel. The potential function is given by physical laws.

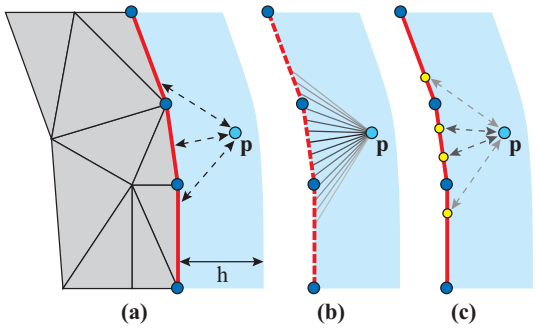


Figure 4: (a) 2D cut through a 3D mesh. Fluid particles within interaction range h from the surface interact with the triangles (shown in red). (b) The convolution integral sums up the contributions of infinitesimal parts of the triangles properly weighted. (c) Interactions with Gaussian particles (yellow) approximate the convolution in an optimal way.

4.3 Gaussian Boundary Particles

Our idea to solve the convolution integral is to use Gauss quadrature rules [17]. For the interaction potential of a particle $\mathbf{p}$ with a single triangle t we get

$$U(\mathbf{p}, t) = \int_{\mathbf{x} \in t} U(\mathbf{p} - \mathbf{x}) d\mathbf{x} \tag{12}$$

$$\approx A \sum_i w_i U(\mathbf{p} - \mathbf{x}_i), \tag{13}$$

where A is the surface area of t, $\mathbf{x}_i$ the sampling points and w_i their weights according to a chosen quadrature rule. We use the seven point rule which has convergence order $O(L^6)$ with respect to the triangle size L. (see Fig. 5(a) and Tab. 1). These sampling points can be interpreted as boundary particles, which are placed and weighted according to the chosen Gauss quadrature rule. The weighted summation of their potentials approximates the convolution of the potential over the domain of the boundary triangle in an optimal way.

Although the seven point rule yields good approximations of the convolution integral, triangles that are large in comparison to the interaction range of the surface would induce a poor sampling of the boundary field. Therefore, we subdivide the boundary triangle until a sufficient sampling rate is provided. We define a threshold for the maximal acceptable distance between boundary particles. This threshold is chosen relative to the maximal interaction radius of the fluid particles and can be regulated by the user. The boundary particles are generated by subdividing the triangle domain and by application of the Gauss quadrature rule to the resulting triangles (see Fig. 5(b)). This has to be done at every time step, because triangles on the boundary are moved and deformed. Therefore, an efficient scheme is needed. We compute the relative vectors from the triangle nodes (shown in blue) to the boundary particles (shown in red) only once because they are the same for all subdivision triangles. These vectors are then added to the blue nodes to generate the complete set of boundary particles. Analog to positions, the velocities of boundary particles are interpolated from the velocities of the triangle nodes.

Now that we have replaced the triangulated surface by a set of particles, the problem of triangle-particle interaction reduces to particle-particle interaction. We can, thus, use SPH-based approaches to approximate the boundary conditions stated in Sections 3.2, 3.3 and 3.4.

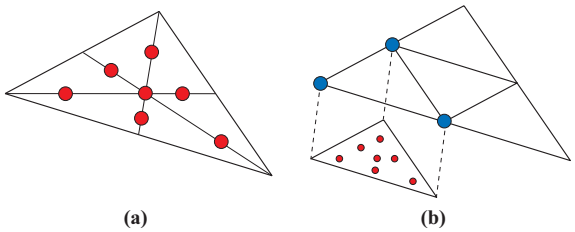


Figure 5: (a) Boundary particles on a triangle according to the seven point rule. (b) Large triangles are subdivided and boundary particles are generated for each resulting triangle.

4.4 Boundary Repulsion and Adhesion

The no-penetration condition stated in Sec. 3.2 prevents fluid particles from penetrating the solid object. Monaghan [16] uses a Lennard-Jones-like force to generate repulsive forces which approximate the no-penetration condition. We propose a Lennard-Jones-like force that models both repulsion and adhesion to the contact surface. We define the force acting on

Point	Barycentric coordinates	Weights
1	$(1/3, 1/3, 1/3)$	9/40
2	(a, b, b)	e
3	(b, a, b)	e
4	(b, b, a)	e
5	(c, d, d)	f
6	(d, c, d)	f
7	(d, d, c)	f

Table 1: *Barycentric coordinates and weights of the seven point Gauss quadrature rule for triangles, where* $a = 0.05971587$*,* $b = 0.47014206$*,* $c = 0.79742699$*,* $d = 0.10128651$*,* $e = (155 + \sqrt{15})/1200$ *and* $f = (155 - \sqrt{15})/1200$*.*

particle $\mathbf{p}$ due to triangle t by the convolution

$$\mathbf{f}^{ra}(\mathbf{p}, t) = \int_{\mathbf{x} \in t} \tau^{ra}(|\mathbf{p} - \mathbf{x}|)\, d\mathbf{x}. \tag{14}$$

The traction τ^{ra} is dependent on the distance of the surface element from the particle $\mathbf{p}$ and has unit *force per area* in order to yield a force when integrated over the triangle. To model repulsion and adhesion, we use the following traction function

$$\tau^{ra}(r) = \begin{cases} k\frac{(h-r)^4 - (h-r_0)^2(h-r)^2}{h^2 r_0 (2h - r_0)} & \text{if } r < h \\ 0 & \text{otherwise} \end{cases}, \tag{15}$$

where h is the interaction range and k controls the stiffness of the interaction. The traction has an order four repulsion term and an order two attraction term. It is designed to be zero for $r = r_0$ which is the preferred distance of fluid particles from the interface. The fact that for $r = 0$ the traction is finite ($\tau^{ra}(0) = k$) and that both, traction and first derivative vanish for $r = h$ are important for robust real time simulations. Using Gaussian boundary particles, the force acting on a particle $\mathbf{p}$ is computed as

$$\mathbf{f}^{ra}(\mathbf{p}) \approx \sum_i A_i \sum_j w_{ij} \tau^{ra}(|\mathbf{p} - \mathbf{x}_{ij}|), \tag{16}$$

where i iterates over all triangles within distance h of particle $\mathbf{p}$. For each triangle the contributions of its boundary particles are summed up according to equation (13).

4.5 Boundary Friction

The no-slip condition (Eq. (5)) can be approximated by including the boundary particles into the viscosity evaluation of the SPH particles [16]. We use the normalized kernel W^{visc} proposed in [13] for viscosity computations. To evaluate the viscosity force $\mathbf{f}^{visc}(\mathbf{p})$ on a fluid particle, the velocities of the boundary particles have to be interpolated from the velocity of mesh nodes (see Sec. 4.3). The traction τ^{visc} depends on the velocity $\mathbf{v}_b$ of the boundary particle, the $\mathbf{v}_p$ of the fluid particle and the distance r between them

$$\tau^{visc}(r) = \mu(\mathbf{v}_b - \mathbf{v}_p)\nabla^2 W_{visc}(r, h), \tag{17}$$

where the scalar μ controls the boundary viscosity and

$$\nabla^2 W_{visc}(r, h) = \begin{cases} \frac{45}{\pi h^6}(h - r) & \text{if } 0 \le r \le h \\ 0 & \text{otherwise.} \end{cases} \tag{18}$$

The kernel W_{visc} is designed such that its Laplacian $\nabla^2 W_{visc}$ takes the linear form above, but satisfies the normalization criterion on the kernel itself. The normalization warrants second order interpolation convergence. The numerical approximation of the convolution integral over the triangle surface defines the final form of the viscosity force

$$\mathbf{f}^{visc}(\mathbf{p}) = \sum_i A_i \sum_j w_{ij} \tau^{visc}(|\mathbf{p} - \mathbf{x}_{ij}|). \tag{19}$$

4.6 Actio = Reactio

So far, we have applied forces to fluid particles only. However, according to Newton's Third Law, proper reaction forces need to be applied to the deformable solid as well. The force contributions of boundary particles have to be distributed among the boundary triangle vertices so they can be picked up by the simulator of the deformable object. Bridson et al. [26] solve a similar problem in the context of cloth simulation. To resolve vertex-triangle collisions, an impulse is applied to the colliding vertex. Then, a distribution scheme is used to compute the corresponding reaction impulses for the three vertices of the triangle. We use the same scheme to distribute the forces to the vertices of the triangle surface. Given the force contribution $\mathbf{f}_b$ computed for one boundary particle we compute the force contributions to the triangle nodes and the fluid particle as

$$\mathbf{f}_k^{triangle} = \frac{2 w_k \mathbf{f}_b}{1 + w_1^2 + w_2^2 + w_3^2} \tag{20}$$

$$\mathbf{f}^{particle} = \frac{-2\mathbf{f}_b}{1 + w_1^2 + w_2^2 + w_3^2}, \tag{21}$$

where the w_k are the barycentric coordinates of the boundary particle with respect to the triangle and $k \in (1 \ldots 3)$. According to [26] this distribution scheme provides continuity across triangle boundaries and introduces appropriate torques for off-center interactions. However, the scheme is not completely error free. Force magnitudes can get amplified – at most by a factor of $8/7$ – at the triangle center. However, this error did not cause any artifacts or stability problems in our simulations.

5 Implementation

At every time step of the solid and fluid simulator, the following five steps are executed:

1. **Surface triangle extraction**: Boundary triangle references are stored in a flat list.
2. **Particle grid hashing**: A grid index on the fluid particles is created.
3. **Neighbor search**: For each boundary triangle a list of possible fluid interaction partners is generated.
4. **Boundary sampling**: For every boundary triangle with possible interaction partners, boundary particles are generated.
5. **Interaction computation**: For every interaction pair, composed of a boundary particle and a fluid particle, forces are computed and applied to related triangles and fluid particles.

Processing the five phases one after the other would have a negative impact on storage requirements. Neighbor references and boundary particles for all triangles would have to

be stored at the same time. If the computations of steps three to five are grouped around single triangles, only data relevant for the current triangle has to be stored at a time (Fig. 6).

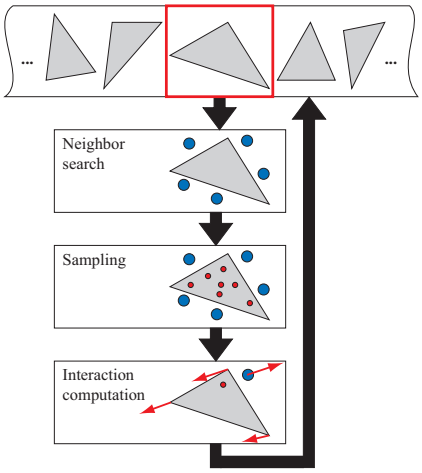


Figure 6: Algorithm overview: Triangles are processed separately. This avoids the storage of fluid particle neighbor lists and boundary particles for all triangles simultaneously.

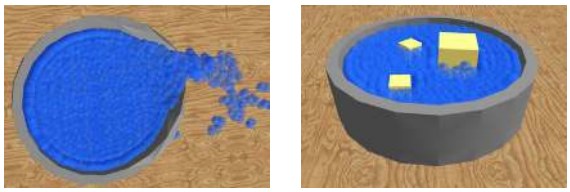
The output of step 3 is a list, containing all fluid particles within interaction range h of a triangle t. To speed up the search for these particles we use a regular grid with spatial hashing [27]. There is a trade-off between computation time for the neighbor search and the quality of the neighbor list. We extend the axes aligned bounding box (AABB) of t along all axes about the interaction range. Then, we query all grid cells intersecting the extended box. We also tested tighter queries which generate fewer neighbor candidates but their increased time complexity was not compensated by the reduced cost of interaction computations.

In step 4, boundary particles are only generated for those triangles that have fluid particle neighbors. The boundary particles for a triangle t are kept only temporarily for the interaction computation. After t is processed, they are discarded. In this step, positions and velocities are interpolated from the triangle nodes for each boundary particle.

To compute interaction forces in step 5 we iterate over all the boundary particles of a triangle. For each fluid particle within the interaction radius of the boundary particle, we compute the interaction forces as described in Sec. 4 and distribute them among the fluid particle and the triangle nodes according to Eqns. (20) and (21).

6 Results

All experiments described in this section have been performed on an AMD Athlon 1.8 GHz PC with 512 MB RAM and a GeForce Ti 4400 graphics card with 128 MB RAM. Note that most of the simulations are recorded in a real-time interactive environment. Thus, we cannot afford several seconds or even minutes per frame for the reconstruction and rendering of the free fluid surface as in off-line simulations [14, 15] which explains the simplistic renderings of the fluids.



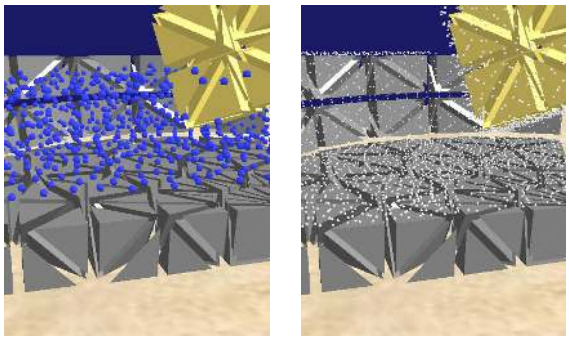
(a) (b)

Figure 7: (a) When the user pulls the pool wall, water flows out. (b) Boxes float on the water surface.

6.1 A Pool Filled with Water

To demonstrate the stability of our model in connection with concave surfaces, we filled a pool composed of 800 tetrahedral elements with 2000 fluid particles (see Fig. 7(a)). The simulation runs at 20 frames per second. By pulling the pool wall, the user indirectly influences the water. The generated waves, in turn, deform the pool walls. Deformable boxes float freely on the water surface (see Fig. 7(b)). Fig. 8 shows the fluid and boundary particles used in the simulation.

6.2 Floating Boxes



(a) (b)

Figure 8: A box floats in a pool: (a) The fluid particles are shown in blue. (b) For the interaction computation virtual boundary particles (white) are placed on the surfaces of the deformable objects.

We dropped an additional large box into the pool (see Fig. 1). When it touches the water, it emits a wave that hits the pool boundary and causes it to fracture. This scene demonstrates the interplay of various physical phenomena provided by the fluid simulator, the solid simulator and the interaction model.

6.3 Simulation of Blood Vessels

An important application of our method is the simulation of bleeding during virtual operations. Our simulation of a blood vessel is a first step into this direction. We simulate the flow of 3000 particles through a virtual vessel, consisting of a deformable mesh composed of 560 tetrahedra. The simulation took about 70 ms per time step. Fig. 9 shows the resulting blood flow. The velocity of the fluid particles is color coded visualizing the friction of the fluid with the boundary.

In the experiment shown in Fig. 10, we turned on fracture of the Finite Element mesh. Now, the vessel is torn open when the elastic stresses caused by blood pressure exceed the material threshold. The free surface of the particle system is

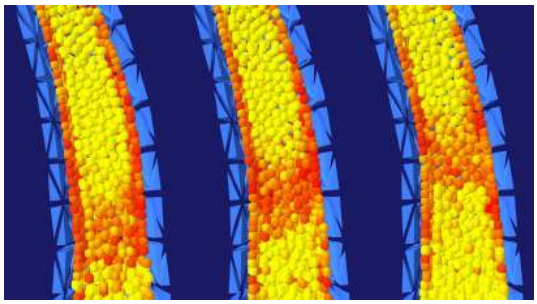


Figure 9: Blood flow through a vessel. The image shows subsequent time slices of an interactive animation. The velocity of the fluid particles is color coded. Yellow colored particles are fast, while red ones are slow. Pulsation waves and viscosity at the vessel boundary can be observed.

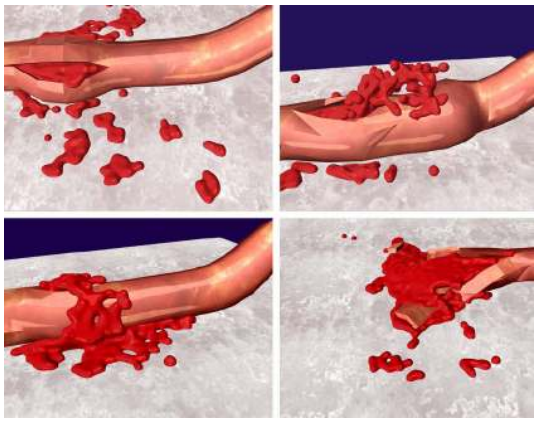


Figure 10: Vessel injury. The Finite Element mesh fractures due to pressure forces in the blood stream.

rendered using the Marching Cubes algorithm. The animation of the mesh and the particles are possible in real time at 60 ms per time step, while surface reconstruction took about half a second per frame. On today's hardware only a limited number of fluid particles can be simulated in real-time which yields a relatively coarse fluid surface.

7 Conclusion

We have presented a new method for the simulation of interactions of deformable solids with fluids. Our interaction model simulates repulsion, adhesion and friction near the fluid-solid interface. The smoothness of the force fields is important for the stability of the simulation. The core idea to get smooth interaction fields is to place boundary particles onto the surface triangles according to Gauss quadrature rules. This idea might be useful in other graphic domains as well. We mentioned the application to modeling with implicit surfaces. Character skinning is another application where bulges or knees are known problems in regions where several close bones meet.

We demonstrated the usability of our method in an interactive simulation environment with several scenes. A difficulty in connection with the interactive simulation of fluids is the extraction and rendering of a plausible fluid surface in real time. Thus, ongoing work focusses on fast algorithms for surface reconstruction.

Acknowledgements

This project was funded by the Swiss National Commission for Technology and Innovation (KTI) project no. 6310.1 KTS-ET.

References

[1] D. Terzopoulos, J. Platt, A. Barr, and K. Fleischer. Elastically deformable models. In *Computer Graphics Proceedings*, Annual Conference Series, pages 205–214. ACM SIGGRAPH 87, July 1987.

[2] David Baraff and Andrew Witkin. Large steps in cloth simulation. In *Proceedings of SIGGRAPH 1998*, pages 43–54, 1998.

[3] Mathieu Desbrun, Peter Schrder, and Alan H. Barr. Interactive animation of structured deformable objects. In *Graphics Interface '99*, 1999.

[4] Doug L. James and Dinesh K. Pai. Artdefo, accurate real time deformable objects. In *Computer Graphics Proceedings*, Annual Conference Series, pages 65–72. ACM SIGGRAPH 99, August 1999.

[5] Gilles Debunne, Mathieu Desbrun, Marie-Paule Cani, and Alan Barr. Dynamic real-time deformations using space & time adaptive sampling. In *Computer Graphics Proceedings*, Annual Conference Series, pages 31–36. ACM SIGGRAPH 2001, August 2001.

[6] E. Grinspun, P. Krysl, and P. Schrder. CHARMS: A simple framework for adaptive simulation. In *ACM Transactions on Graphics*, volume 21, pages 281–290. ACM SIGGRAPH 2002, August 2002.

[7] Matthias Mller, Julie Dorsey, Leonard McMillan, R. Jagnow, and B. Cutler. Stable real-time deformations. *Proceedings of 2002 ACM SIGGRAPH Symposium on Computer Animation*, pages 49–54, 2002.

[8] W. T. Reeves. Particle systems — a technique for modeling a class of fuzzy objects. *ACM Transactions on Graphics 2(2)*, pages 91–108, 1983.

[9] Mathieu Desbrun and Marie-Paule Cani. Smoothed particles: A new paradigm for animating highly deformable bodies. In *6th Eurographics Workshop on Computer Animation and Simulation '96*, pages 61–76, 1996.

[10] J.J. Monaghan. Smoothed particle hydrodynamics. *Annu. Rev. Astron. Physics*, 30:543, 1992.

[11] Mathieu Desbrun and Marie-Paule Cani. Space-time adaptive simulation of highly deformable substances. Technical report, INRIA Nr. 3829, 1999.

[12] D. Stora, P. Agliati, M. P. Cani, F. Neyret, and J. Gascuel. Animating lava flows. In *Graphics Interface*, pages 203–210, 1999.

[13] Matthias Mller, David Charypar, and Markus Gross. Particle-based fluid simulation for interactive applications. *Proceedings of 2003 ACM SIGGRAPH Symposium on Computer Animation*, pages 154–159, 2003.

[14] Simon Premoze, Tolga Tasdizen, James Bigler, Aaron Lefohn, and Ross T. Whitaker. Particle-based simulation of fluids. *Eurographics*, 22(3):401–410, 2003.

[15] Olivier Génevaux, Arash Habibi, and Jean-Michel Dischler. Simulating fluid-solid interaction. In *Graphics Interface*, pages 31–38. CIPS, Canadian Human-Computer Commnication Society, A K Peters, June 2003. ISBN 1-56881-207-8, ISSN 0713-5424.

[16] J. J. Monaghan, M. Thompson, and K. Hourigan. Simulation of free surface flows with sph. *ASME Symposium on Computational Methods in Fluid Dynamics*, 1994.

[17] C. Pozrikidis. *Numerical Computation in Science and Engineering*. Oxford Univ. Press, NY, 1998.

[18] T. J. Chung. *Applied Continuum Mechanics*. Cambridge Univ. Press, NY, 1996.

[19] J. F. O'Brien and J. K. Hodgins. Graphical modeling and animation of brittle fracture. In *Proceedings of SIGGRAPH 1999*, pages 287–296, 1999.

[20] Jos Stam. Stable fluids. In *Proceedings of the 26th annual conference on Computer graphics and interactive techniques*, pages 121–128. ACM Press/Addison-Wesley Publishing Co., 1999.

[21] N. Foster and R. Fedkiw. Practical animation of liquids. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pages 23–30. ACM Press, 2001.

[22] D. Enright, S. Marschner, and R. Fedkiw. Animation and rendering of complex water surfaces. In *Proceedings of the 29th annual conference on Computer graphics and interactive techniques*, pages 736–744. ACM Press, 2002.

[23] J. Blinn. A generalization of algebraic surface drawing. *ACM Transactions on Graphics*, 1(3):235–256, 1982.

[24] J. Bloomenthal. *Skeletal Design of Natural Forms*. PhD thesis, University of Calgary, Canada, 1995.

[25] A. Sherstyuk. Fast ray tracing of implicit surfaces. *In Implicit Sufaces '98*, pages 145–153, 1998.

[26] R. Bridson, R. Fedkiw, and J. Anderson. Robust treatment of collisions, contact and friction for cloth animation. In *ACM Transactions on Graphics*, volume 21, pages 594–603. ACM SIGGRAPH 2002, August 2002.

[27] M. Teschner, B. Heidelberger, M. Mller, D. Pomeranerts, and M. Gross. Optimized spatial hashing for collision detection of deformable objects. In *Proc. Vision, Modeling, Visualization VMV*, pages 47–54, 2003.