Constraint Fluids

Kenneth Bodin, Claude Lacoursière, Martin Servin

***Abstract*—We present a fluid simulation method based on Smoothed Particle Hydrodynamics (SPH) in which incompressibility and boundary conditions are enforced using holonomic kinematic constraints on the density. This formulation enables systematic multiphysics integration in which interactions are modeled via similar constraints between the fluid pseudo-particles and impenetrable surfaces of other bodies. These conditions embody Archimede's principle for solids and thus buoyancy results as a direct consequence. We use a variational time stepping scheme suitable for general constrained multibody systems we call SPOOK. Each step requires the solution of only one Mixed Linear Complementarity Problem (MLCP) with very few inequalities, corresponding to solid boundary conditions. We solve this MLCP with a fast iterative method. Overall stability is vastly improved in comparison to the unconstrained version of SPH, and this allows much larger time steps, and an increase in overall performance by two orders of magnitude. Proof of concept is given for computer graphics applications and interactive simulations.**

I. Introduction

PARTICLE based methods for simulating Navier-Stokes equations for fluids were introduced in computer graphics by Desbrun and Cani [**?**], who used Smoothed Particle Hydrodynamics (SPH) for animating soft objects. Further work has demonstrated animation of lava flows [2], interactive fluid simulation [3], blood flow [4] and representation of fluids in medical simulators [5], fluid-fluid interaction [6], and sand-fluid interaction [7]. Extensions of SPH and the moving least squares method have been used for animating elastic and plastic materials [8].

In SPH, the fields of partial differential equations (PDEs) are replaced by weighted volumetric approximations. These are the smoothed particles which interact with their neighbors via manybody potentials corresponding to the various terms appearing in the original PDE. Each of these potentials is computed as a weighed average over the smoothed particles. This makes the dynamical system look much like molecular dynamics, though the interactions are no longer pairwise only.

SPH was first introduced in astrophysics independently by Lucy [9] and Gingold and Monaghan [10], to simulate interstellar flows. But SPH and other particle methods are now widely spread in various fields of science and engineering [11]–[13].

Point and particle methods are simple and versatile making them attractive for animation purposes. They are also common across computer graphics applications, such as storage and visualization of large-scale data sets from 3D-scans [14], 3D modeling [15], volumetric rendering of medical data such as CT-scans [16], and for voxel based haptic rendering [17], [18]. The lack of explicit connectivity is a strength because it results in simple data structures. But it is also a weakness since the connection to the triangle surface paradigm in computer graphics requires additional computing, and because neighboring particles have to be found. The overhead in reconstructing polygon surfaces has motivated the development of specific techniques for rendering of points and particles. Important steps include Reeves' [19] original introduction of particles in graphics, the pioneering work of Levoy and Whitted [20] on using points as display primitives, the introduction of surfels [21] and a broad range of techniques, now often referred to as point based rendering and point splatting (see [22] for a review).

Despite good progress on the modeling front, material particle methods like SPH are subject to performance and stability bottlenecks. This comes from generalized use of explicit time integration methods which have serious limitations with regards to the size of time step, and poor performance when applied to stiff systems.

For the case of a fluid, such as a gas or a liquid, the speed of sound is closely related to the incompressibility, since it determines the propagation speed of impulses caused by external forces and boundary conditions, as well as the dynamics of non-stationary flow. For traditional SPH with pairwise temporal propagation of forces, we can estimate the required minimum time step Δt for stability from the Courant-Friedrichs-Lewy condition [23], as $\Delta t < h/v$, where h is the spatial resolution and interaction length, and v is the force propagation speed. For water, compression waves travel at the speed of sound which is nearly 1500 m/s. This requires time steps in the range of 10^{-7} to 10^{-6} seconds for simulations with a length scale resolution of 1 to 10 cm. That is between four and six orders of magnitude smaller than real-time graphics requirements. The effective compressibility is roughly six order of magnitudes larger for typical time steps of ten milliseconds. The incompressibility of water—and of air for that matter—are fundamental to the qualitative aspects of the flow. This means that simulated motion cannot even be plausible to the naked eye for the accessible speed of sound. Even for offline simulation, a lot of noise

K. Bodin and C. Lacoursière, are with the High Performance Computing Center, Umeå University, SE-901 87 Umeå, Sweden, and Algoryx Simulation AB, Umeå, Sweden e-mail: kenneth@algoryx.se, claude@hpc2n.umu.se.

M. Servin is with the Dept., of Physics, Umeå University, SE-901 87 Umeå, Sweden, and Algoryx Simulation AB, Umeå, Sweden e-mail: martin.servin@physics.umu.se.

Manuscript received Sept X, 2009; revised X Y, 2009.

and inaccuracies can accumulate over three orders of magnitude, and this motivates looking for simple implicit integration methods. A similar class of problems arises in constrained multibodies, especially with regards to contact problems. Penalty methods for these are equivalent to Sph methods for fluids, since both rely exclusively on explicit force computations, but these are notoriously difficult to tune to achieve stability. Sph can be improved much in the same way as rigid body methods by projection on a divergence free velocity field, or, as presented in this paper, by means of a direct multibody constraint on the mass density. Either approach corresponds to the imposition of a global constraint on the system instead of locally as is the case for forced based methods.

In the rest of this paper, we first review some related work in § II before introducing the Sph approximation scheme in § III. Constrained dynamics and the Spook integration method is explained in § IV, leading to the specific incompressibility constraint for Sph in § V and for boundary conditions in § VI. The solution method for the resulting linear problems is described in § VII-A, and the integration with multibody systems in § VII-B. The overall algorithm is spelled out in § VIII and results are presented in § IX, followed by a brief conclusion in § X.

II. Related work

A variety of methods for fluid simulations exist, including finite element and finite difference methods, particle-in-cell methods, level set methods, Lattice-Boltzmann methods, various types of particle methods, and more. Most of these have also found their way into computer graphics but we will restrict our review of related work to particle methods. For a recent overview of other methods, please refer to [24].

An important contribution came from Cummins and Rudman, who developed a method for projecting velocities onto a divergence free velocity field resulting in incompressible flow [25]. Through mass conservation and the density continuity equation this also results in conservation of density and fluid volume. A related approach, the Moving Particle Semi-Implicit (MPS) method [26], has also found applications in computer graphics [27]. A recent comparison of methods for incompressible and weakly incompressible Sph concludes that these methods are mutually consistent, and that they also are in good agreement with empirical data [28].

A particle volume constraint approach [29], not distant from the work presented here, uses the Shake integration method of molecular dynamics [30]. This requires nearly exact roots of the nonlinear constraint equation, using Newton-Raphson iterations. A predictor-corrector incompressible Sph approach (PCISPH) was recently introduced in graphics [31], but has not yet gone through the standard benchmark tests of computational fluids dynamics. In PCISPH the density is linearly expanded in the particle positions and the resulting linear system is solved by means of Jacobi iterations, i.e. particles are moved around slightly to achieve incompressibility. As with many position based methods, PCISPH is rather easy to implement and gives good drift free constraint satisfaction and convergence. However, for multiphysics simulations the position projection approach only integrates well with other position based methods (e.g. [32]), while it does not integrate well with the more established force and velocity based methods that we use in the current paper. Position methods typically also deliver imprecise and noisy force estimates, since forces are two indices (time derivatives) away from the positions, making these methods impractical in applications where forces are important, e.g., haptic surgery training simulators. More seriously, position projection methods do not automatically satisfy Newton's third law, and thus risk violating conservation of linear and angular momentum. When position methods are explicitly designed to conserve momentum, by including inertia, they actually reinvent force dynamics and linear implicit integration.

None of the aforementioned methods provide the combination of near incompressibility, stability, conservation properties, computational efficiency and systematic integration of multiphysics systems, motivating the present work.

III. Smoothed Particle Hydrodynamics

In the Sph approximation, particles carry a weighted average field value. This is smoothed over the neighboring particles. Details of this approximation are covered in monographs and review articles [11]–[13].

For any scalar field A, the value at particle i positioned at $\mathbf{r}_i$ is computed as

$$A_i = \sum_j m_j \frac{A_j}{\rho_j} W_{ij}(r_{ij}, h), \tag{1}$$

where j is over all neighboring particles, including i, found within the smoothing length h. This length includes the compact support of the kernel function $W_{ij} = W(r_{ij}, h)$, where $r_{ij} = \left|\mathbf{r}_i - \mathbf{r}_j\right|$, m_j is the mass of particle j, and ρ_j is the smoothed density of particle j computed from

$$\rho_i = \sum_j m_j W_{ij}. \tag{2}$$

To obtain a good approximation, the kernel itself should be normalized on each particle i

$$1 = \sum_j m_j \frac{1}{\rho_j} W_{ij}. \tag{3}$$

The beauty of Sph is that field differentials such as $\boldsymbol{\nabla} A$ and $\nabla^2 A$, that occur in Navier-Stokes equation are easily computed using Eqn. (1), since the differential operator only acts on the kernel function and can be calculated analytically,

$$\boldsymbol{\nabla}_i A_i = \sum_j m_j \frac{A_j}{\rho_j} \boldsymbol{\nabla}_i W_{ij}, \tag{4}$$

and

$$\nabla_i^2 A_i = \sum_j m_j \frac{A_j}{\rho_j} \nabla_i^2 W_{ij}, \tag{5}$$

for example. The Navier-Stokes' equations of a Newtonian viscous fluid read

$$\rho \frac{D\mathbf{v}}{Dt} = -\boldsymbol{\nabla} p + \mu \nabla^2 \mathbf{v} + \rho \mathbf{g}, \tag{6}$$

where ρ is the fluid density, p the pressure, $\mathbf{g}$ the gravitational field, $\mathbf{v}$ the flow velocity of the fluid, and D/Dt is the substantial derivative. The LHS of Eqn. (6) is the change in momentum, and the RHS corresponds to the total force responsible for this change. In Sph, a Lagrangian particle moving with the fluid has an acceleration given by $D\mathbf{v}/Dt$, so by multiplying with m_i/ρ, we can rewrite Eqn. (6) to obtain an equation of motion for the particle,

$$m_i \frac{\mathrm{d}\mathbf{v}_i}{\mathrm{d}t} = \sum_j \left(\mathbf{f}_{ij}^p + \mathbf{f}_{ij}^v \right) + m_i \mathbf{g}, \tag{7}$$

with a pressure force $\mathbf{f}_{ij}^p$ and a viscosity force $\mathbf{f}_{ij}^v$ contributed from nearby particles j within the range h of the kernel function. Using Eqns. (4)–(5) in Eqn. (6), and symmetrization to fulfill Newton's third law, we can derive expressions for these forces. First, the symmetric pairwise pressure force is approximated as

$$\mathbf{f}_{ij}^p = -m_i m_j \left(\frac{p_i}{\rho_i^2} + \frac{p_j}{\rho_j^2} \right) \boldsymbol{\nabla}_i W_{ij}, \tag{8}$$

and the symmetric viscosity force as

$$\mathbf{f}_{ij}^v = \mu \frac{m_i m_j}{\rho_i \rho_j} \left(\mathbf{v}_j - \mathbf{v}_i \right) \cdot \hat{\mathbf{r}}_{ij} \nabla_i^2 W_{ij}. \tag{9}$$

For small density variations the pressure can be approximated from the Tait equation of state [33],

$$p_i = \frac{\rho_0 c_s^2}{\gamma} \left(\left(\frac{\rho_i}{\rho_0} \right)^{\gamma} - 1 \right) \tag{10}$$

where c_s is the sound velocity in the fluid and ρ_0 is the target fluid density, which is 1000 kg/m^3 for water. A model with $\gamma = 1$ is often referred to as pseudo compressible Sph [?], [3], whereas models with $\gamma = 7$ are referred to as weakly incompressible Sph [34]. Obviously, the latter choice will give a much stiffer equation of state but alas the achievable incompressibility is still entirely limited by the size of the time step.

As we show in this paper, an alternative and more efficient way of expressing the equation of state is to formulate it as a constraint, that is, $g = \left(\frac{\rho}{\rho_0} - 1 \right) = 0$, so that the density, ρ, is constrained to the reference density ρ_0. Therefore we now recapitulate some theory of constrained dynamics and present a new numerical integrator suitable for this system.

IV. Constrained dynamics

In the following we introduce Spook, a novel numerical integrator for constrained systems [35] formulated in descriptor form. This requires an explicit computation of the Lagrange multipliers corresponding to the constraint forces. This method is based on a physically motivated perturbation of the Lagrangian of the mechanical systems, i.e., not on the equations of motion themselves. The perturbation includes both relaxation and dissipation parameters. The former protects against ill-conditioning resulting from degenerate constraint configurations [36], and the latter provides for constraint stabilization. The perturbed system converges uniformly to the exact constrained system, and this can be proven rigorously [37], [38]. The perturbation parameters do not depend explicitly on the time step, a feature shared by some [36], [39] but not all other [40] stabilization techniques. The integrator is based on a discrete-time formulation of the variational principle [30] and an approximation of the constraint equations. We use the simple linear approximation but the quadratic correction is easy to add, though not strictly necessary. For a given time step, the stability of the integrator is limited by the constraint curvature and the value of the dissipation parameter. The time stepping scheme is a variant of what is widely known as an *impulse based* simulation.

A constrained mechanical system with generalized n-dimensional coordinates $\mathbf{q}$ is one that is restricted to move on the surface defined by $\mathbf{g}(\mathbf{q}) = 0$. Here, $\mathbf{g} : \mathbb{R}^n \mapsto \mathbb{R}^m, m \leq n$, is the indicator function which is assumed to be continuously differentiable with $m \times n$ Jacobian matrix $\mathbf{G} = \partial \mathbf{g}/\partial \mathbf{q}$. This is the holonomic scleronomic case. Non-holonomic cases including inequalities and velocity constraints are not covered here but are straight forward extensions.

In the context of classical mechanics, explicit constraints on the motion may be considered as limits of strong potentials as discussed at length in several classic texts [41], [42], and these potentials are those acting on short time and length scales which are of no interest in the present context. Penalty methods use strong potentials directly but the parametrization is usually difficult to map to physical quantities since they are intimately coupled to the choice of integration method. But at the theoretical level at least, the limit of infinite penalty is well defined as long as there is some dissipation along the constraint restoration force [37], since otherwise, wild oscillations could appear as the force strength increases [38]. Constraints $\mathbf{g}(\mathbf{q}) = 0$ sum up the effect of physical phenomena at time and length scales that are much smaller than those we are interested in, i.e., very high frequency and low amplitude oscillations.

If we consider a small constant positive scalar $\epsilon > 0$, we can define the potential function $U_\epsilon(\mathbf{q}) = \frac{1}{2\epsilon} \mathbf{g}^T \mathbf{g}$ producing the force

$$-\frac{\partial U_\epsilon}{\partial \mathbf{q}} = -\frac{1}{\epsilon} \mathbf{G}^T \mathbf{g}(\mathbf{q}). \tag{11}$$

This is just a nonlinear spring force acting on the displacement from equilibrium where $\mathbf{g}(\mathbf{q}) = 0$. If we consider the limit of $\epsilon \to 0$, the system has finite energy only if $\mathbf{g}_\epsilon(\mathbf{q}(t)) \to 0$ for all times t. Conversely, if the energy is bounded along any converging sequence $\{\epsilon_k\}_{k=0}^{\infty}$ with $\epsilon_k \to 0$, the penalty potential remains bounded and $U_\epsilon \to 0$, which means that $(1/\epsilon)\mathbf{g}^T\mathbf{g}/2 \to 0$ as well. Introducing auxiliary variable $\bar{\boldsymbol{\lambda}} = -(1/\epsilon)\mathbf{g}(\mathbf{q})$, the constraint force is $\mathbf{G}^T\bar{\boldsymbol{\lambda}}_\epsilon$. Despite the fast oscillations mentioned previously, $\bar{\boldsymbol{\lambda}}_\epsilon$ converges in the weak sense meaning that the time integral $\int_{t-\sigma}^{t+\sigma} \mathrm{d}s\bar{\boldsymbol{\lambda}}_\epsilon$ converges uniformly for an arbitrary small $\sigma > 0$. This is used in the time discretization below. But damping will work equally well and this can be done with a force of the form $-(\tau/\epsilon)\mathbf{G}^T\mathbf{G}\dot{\mathbf{q}} = (\tau/\epsilon)\mathbf{G}^T\dot{\boldsymbol{\beta}}_\epsilon$, with $\tau \geq 0$ and the auxiliary variable defined implicitly as $\epsilon\dot{\boldsymbol{\beta}}_\epsilon + \tau\mathbf{G}(\mathbf{q})\dot{\mathbf{q}} = 0$. When the two forces are added, we write $\boldsymbol{\lambda}_\epsilon = \bar{\boldsymbol{\lambda}}_\epsilon + \dot{\boldsymbol{\beta}}_\epsilon$. Dropping the ϵ subscript henceforth, the standard equations of motion for classical mechanics become

$$\begin{aligned} \mathbf{M}\ddot{\mathbf{q}} + \dot{\mathbf{M}}\dot{\mathbf{q}} - \mathbf{G}^T\boldsymbol{\lambda} &= f \\ \epsilon\boldsymbol{\lambda} + \mathbf{g}(\mathbf{q}) + \tau\mathbf{G}(\mathbf{q})\dot{\mathbf{q}} &= 0, \end{aligned} \tag{12}$$

where $\mathbf{M}$ is the systems mass matrix. These differential algebraic equations (DAEs) have index 1 [39] for any $\epsilon, \tau \neq 0$. In the limit where $\epsilon, \tau \to 0$, the trajectory $\mathbf{q}(t)$ converges uniformly if the initial conditions satisfy $\mathbf{g}(\mathbf{q}(0)) = 0, \mathbf{G}(\mathbf{q}(0))\dot{\mathbf{q}}(0) = 0$ as is well known [38] [43].

If we set $\epsilon = 0$ and $\tau = 0$ directly, we recover the standard index 3 DAEs of constrained mechanical systems. These are much harder to integrate [39]. The point here is that the $\epsilon > 0$ perturbation in Eqn. (12) is the natural physical one which regularizes the index 3 DAEs of motion to something easier to solve, namely, index 1 DAEs. This is in sharp contrast with other forms of regularizations of the index 3 DAEs which are not based on physics. Keeping $\tau > 0$ but setting $\epsilon = 0$ produces an index 2 DAE, but is different from other index reduction techniques [36], [39], [40].

The system Eqn. (12) contains the embryos of both pure penalty and pure constraint formulations. For the former, one simply eliminates $\boldsymbol{\lambda}$ algebraically from the second equation. This is ill-conditioned because it introduces terms of the form ϵ^{-1}. For the latter, we can first set $\epsilon = 0, \tau = 0$, differentiate the second equation twice, solve for $\boldsymbol{\lambda}$, and substitute back in the first equation. But this does not guarantee either $\dot{\mathbf{g}} = 0$ or $\mathbf{g} = 0$ for all times and thus, some form of constraint stabilization is needed at least at the numerical level. That introduces artificial, non-physical parameters which can prove difficult to tune. We avoid this strategy entirely in what follows.

Let us now turn to the problem of discretizing Eqn. (12) in time without loosing any of the good physics it contains.

If you consider leapfrog or Verlet integration for mechanical systems [30], assuming that $\dot{\mathbf{M}} = 0$, you have $\mathbf{M}\mathbf{v}_{k+1} = \mathbf{M}\mathbf{v}_k + \Delta t\mathbf{f}_k$ and $\mathbf{q}_{k+1} = \mathbf{q}_k + \Delta t\mathbf{v}_{k+1}$, where $\mathbf{f}_k$ is the total force. Now, the constraint force magnitude $\boldsymbol{\lambda}$ should be just that which makes $\mathbf{g}(\mathbf{q}_{k+1}) = 0$ and so a suitable first order discretization is then

$$\begin{aligned} \mathbf{M}\mathbf{v}_{k+1} - \mathbf{G}_k^T\boldsymbol{\lambda} &= \mathbf{M}\mathbf{v}_k + \Delta t\mathbf{f}_k \\ \frac{\epsilon}{\Delta t}\boldsymbol{\lambda} + \mathbf{g}(\mathbf{q}_{k+1}) + \tau\mathbf{G}_k\mathbf{v}_{k+1} &= 0 \\ \mathbf{q}_{k+1} &= \mathbf{q}_k + \Delta t\mathbf{v}_{k+1}, \end{aligned} \tag{13}$$

in which a factor of Δt was absorbed in the definition of $\boldsymbol{\lambda}$. But this is still a nonlinear system of equations. In fact, setting $\epsilon = 0$ and $\tau = 0$, Eqn. (13) is identical to the well known SHAKE integrator [30]. We could simply approximate $\mathbf{g}_{k+1} \approx \mathbf{g}_k + h\mathbf{G}_k v_{k+1}$, but that makes the approximation one-sided, something that would break time reversal invariance which is fundamental in physics. We did use that in our previous work [44], [45]. In practical terms, such a one-sided approximation introduces additional numerical dissipation, which might be desirable in some cases. Our approximation of the discrete stepping equations of Eqn. (13) replaces $\mathbf{g}(\mathbf{q}_{k+1})$ with averaged value as discussed above

$$\frac{\epsilon}{\Delta t}\boldsymbol{\lambda} + \frac{1}{4}(\mathbf{g}_{k+1} + 2\mathbf{g}_k + \mathbf{g}_{k-1}) + \tau\mathbf{G}_k\mathbf{v}_{k+1} = 0. \tag{14}$$

The linearization of this is then implicit in $\mathbf{v}_{k+1}$,

$$\mathbf{G}_k\mathbf{v}_{k+1} + \frac{4\epsilon\xi}{\Delta t^2}\boldsymbol{\lambda} = -\frac{4\xi}{\Delta t}\mathbf{g}_{k+1} + \xi\mathbf{G}_k\mathbf{v}_k, \tag{15}$$

with $\xi = 1/(1 + 4\tau/h)$. A complete derivation of Eqn. (14) is based on the discrete time variational principle [46] which is beyond the present scope.

We can isolate $\boldsymbol{\lambda}$ by substituting $\mathbf{G}_k\mathbf{v}_{k+1} = \mathbf{G}_k\mathbf{v}_k + \mathbf{G}_k\mathbf{M}^{-1}\mathbf{G}_k^T\boldsymbol{\lambda} + h\mathbf{G}_k\mathbf{M}^{-1}\mathbf{f}_k$ from the first equation, and linearize $\mathbf{g}(\mathbf{q}_{k+1}) = \mathbf{g}(\mathbf{q}_k) + \Delta t\mathbf{G}_k\mathbf{v}_{k+1}$ to get the Schur complement form

$$\mathbf{S}_\epsilon\boldsymbol{\lambda} = \mathbf{c} - \mathbf{G}\mathbf{M}^{-1}\mathbf{a}, \tag{16}$$

where $\mathbf{S}_\epsilon$ is the Schur complement matrix

$$\mathbf{S}_\epsilon = \mathbf{G}\mathbf{M}^{-1}\mathbf{G}^T + \boldsymbol{\Sigma}. \tag{17}$$

The latter can be interpreted as the inverse inertia seen by the constraint force, and $\boldsymbol{\Sigma}$ is a diagonal regularization term. For the other terms,

$$\begin{aligned} \mathbf{a} &= \mathbf{M}\mathbf{v}_k + \Delta t\mathbf{f}_k \\ \mathbf{c} &= -\frac{4}{\Delta t}\boldsymbol{\Xi}\mathbf{g}_k + \boldsymbol{\Xi}\mathbf{G}_k\mathbf{v}_k \end{aligned} \tag{18}$$

with the definitions

$$\boldsymbol{\Xi} = \mathrm{diag}(\xi_1, \xi_2, \ldots, \xi_{m_h}) \tag{19}$$

$$\boldsymbol{\Sigma} = \frac{4}{\Delta t^2}\,\mathrm{diag}\left(\frac{\epsilon_1}{\xi_1}, \frac{\epsilon_2}{\xi_2}, \ldots, \frac{\epsilon_{m_h}}{\xi_{m_h}}\right). \tag{20}$$

This might look complicated in matrix form but since $\boldsymbol{\Xi}$ and $\boldsymbol{\Sigma}$ are diagonal matrices, operations on these are straight forward. The complete solver and parametrization defined in Eqns. (16)–(20) is what we call SPOOK. That name was chosen because the auxiliary variables $\boldsymbol{\lambda}$ are essentially massless point particles (”ghost particles”) [35] in the Lagrangian formulation. For $\epsilon > 0$ and $\tau > 0$, SPOOK is linearly stable [46].

Observe that for Eqn. (16) there is no problem in the limit $\epsilon \to 0$ numerically, and that the parameter $\tau/\Delta t$ is not critically important, except with regards to stability. This is easy to see by considering a single scalar linear constraint in which case the constraint violation is damped by a factor of $1/(1 + \tau/\Delta t)$ at each time step. In all our simulations presented here, we use $\tau/\Delta t \in (2, 5)$ and ϵ of the order of 10^{-8} to 10^{-3}. When τ is too small, the simulation becomes unstable. When τ is too large, the constraints fail to stabilize quickly. With regards to ϵ, large values introduce elasticity in the constraints but make the linear equations easier to solve both for direct solvers and iterative ones. In the latter case, a large ϵ improves diagonal dominance which helps convergence of Gauss-Seidel methods [47]. It is possible to use $\epsilon = 0$ directly provided the Jacobian matrices G have full row rank and the condition number of matrix $S_{\epsilon=0}$ is moderate.

It is possible to directly relate these parameters to conventional material parameters, as was done in Ref. [45] where a similar approach was taken to mesh-based simulation of deformable solids. The choice also reflects stability versus the mass ranges and stresses in the system, and thus also relates to the effective speed of sound in a fluid. We also showed elsewhere [46] that the error in constraint violation is of order $O(\Delta t^2)$.

We now turn to the specific constraint definition which applies to incompressible fluid flow.

V. A HOLONOMIC MANYBODY MASS DENSITY CONSTRAINT

For an incompressible fluid, the mass density is constant throughout the fluid volume. To enforce this we formulate a manybody constraint g_i for particle i, that constrains the SPH density to the target density of the real fluid, ρ_0,

$$g_i = \frac{\rho_i}{\rho_0} - 1 = \frac{1}{\rho_0}\sum_j m_j W_{ij} - 1 \tag{21}$$

The corresponding kinematic constraint for particle i is,

$$\begin{aligned} \dot{g}_i &= \frac{d}{dt}\left(\frac{1}{\rho_0}\sum_j m_j W_{ij} - 1\right) \qquad (22)\\ &= \frac{1}{\rho_0}\sum_j m_j\ f_{ij}\ \hat{\mathbf{r}}_{ij}\cdot\mathbf{v}_{ij}, \end{aligned}$$

where we have introduced $f_{ij} = \frac{dW_{ij}}{dr_{ij}}$ and $\hat{\mathbf{r}} = \frac{\mathbf{r}}{|\mathbf{r}|}$. The constraint g_i can be interpreted as a weighted manybody distance constraint acting to preserve a weighted average at each particle, i.e., the mass density at each particle position. To the best of our knowledge, the formulation of a density constraint is novel and not to be found elsewhere in the scientific literature.

Following the previous section, and $\dot{\mathbf{g}} = \mathbf{G}\mathbf{v}$, the components of the full Jacobian matrix for all particles, corresponding to the constraint, Eqn. (22) are identified to be,

$$\mathbf{G}_{ij} = -\frac{m_j}{\rho_0} f_{ij}\hat{\mathbf{r}}_{ij}^T \tag{23}$$

$$\mathbf{G}_{ii} = \frac{1}{\rho_0}\sum_k m_k f_{ik}\hat{\mathbf{r}}_{ik}^T \tag{24}$$

The kernel function, W_{ij}, and thus also its derivative, f_{ij}, have compact support, so only particles within the smoothing distance h have non-zero f_{ij}. Therefore, the Jacobian is very sparse for large systems since only the neighboring particles participate in the constraint. Knowing that the Lagrange multipliers $\boldsymbol{\lambda}$ correspond to the constraint forces, $\mathbf{f} = \mathbf{G}^T\boldsymbol{\lambda}$, and with help of Eqns. (23)–(24) we can compute the total constraint force on particle i as,

$$\mathbf{f}_i = \sum_j \left[m_i\boldsymbol{\lambda}_i + m_j\boldsymbol{\lambda}_j\right] f_{ij}\hat{\mathbf{r}}_{ij}. \tag{25}$$

The two-body force $\mathbf{f}_{ij}$ is antisymmetric in ij and thus satisfies Newton's third law and conserves linear momentum. It is a central force acting along $\hat{\mathbf{r}}_{ij}$ and therefore it also conserves angular momentum.

VI. BOUNDARY CONDITIONS

Different models for boundary conditions have been proposed for SPH, e.g., inert boundary particles, per particle non penetration constraints, and penalty potentials (see [12] for a review). The perhaps most common method in the scientific and engineering literature is the boundary particle method, where 2–3 layers of inert particles cover all boundaries and create an SPH consistent boundary condition. A strength of this method is that the normalization condition, Eqn. (3) is satisfied also for particles at the proximity of a boundary. However, this method adds many particles to the simulation, and increases the computational cost. It also requires a rather complicated analysis of the boundaries of the system for positioning the boundary particles. For the purpose of this paper, we use two different methods for 2D and 3D simulation, respectively. In 3D, we use the simplest possible method, a non penetration constraint that prevents the particles from coming closer to a boundary than half the smoothing length, h.

If we consider the smoothed particles as non-penetrable spheres of radius h, a non penetration condition between a particle at position $\mathbf{r}$ with a plane, defined with normal $\mathbf{n}$ and reference point $\mathbf{r}_0$ is just

$$\mathbf{n}\cdot(\mathbf{r} - \mathbf{r}_0) - h/2 \geq 0, \tag{26}$$

and the Jacobian for that is simply $\mathbf{G} = \mathbf{n}^T$. For contacts between particles and rigid bodies, the same logic applies but now, the reference point is moving with the rigid body. Assume the reference point is $\bar{\mathbf{p}}_b$ in the rigid body frame based at $\mathbf{r}_b$, the center of mass of body b. The world coordinates of the contact point $\mathbf{r}_c$ are then

$$\mathbf{r}_c(t) = \mathbf{r}_b(t) + \mathbf{R}_b \bar{\mathbf{p}}_b, \tag{27}$$

and the time derivative of that is

$$\dot{\mathbf{p}}_b = \mathbf{v}_b + \boldsymbol{\omega}_b \times \mathbf{p}_b, \tag{28}$$

where $\mathbf{p}_b = \mathbf{R}_b \bar{\mathbf{p}}_b$, $\mathbf{R}_b$ is the rotation matrix from body b to world coordinates, and $\boldsymbol{\omega}_b$ is the angular velocity of body b expressed in world coordinates. A small amount of algebraic manipulation yields

$$\dot{\mathbf{g}} = \mathbf{G}\begin{bmatrix}\mathbf{v}\\ \mathbf{v}_b\\ \boldsymbol{\omega}_b\end{bmatrix} = \begin{bmatrix}\mathbf{n}^T & -\mathbf{n}^T & -(\mathbf{p}_b \times \mathbf{n})^T\end{bmatrix}\begin{bmatrix}\mathbf{v}\\ \mathbf{v}_b\\ \boldsymbol{\omega}_b\end{bmatrix}. \tag{29}$$

For 2D simulations, we use an SPH boundary field scheme, which to our knowledge also is new. When a particle is found to be within one smoothing length from a boundary, we add a smoothed mass field, $mW(r)$, to the density sum of the particle, Eqn. (2). m is here the particle mass, and W has the same smoothing distance as the particle-particle kernels. This results in an boundary element block in the constraint Jacobian matrix, and in effect an efficient boundary constraint for the particle and a reaction pressure on the boundary. In 2D, we use signed distance fields for detecting and computing contact information. This means that both the distance to the boundary, and the boundary normal, are efficiently available through lookup. The method also allows us to smooth the precomputed normals to greatly improve the boundary model near kinks and corners in the boundary geometry. For our purposes, we find that the method is comparable to, and often more efficient than the recently reported multiple boundary tangent method [48]. We will publish a detailed study elsewhere. Our boundary field method should also lend itself to efficient modeling in 3D simulations.

VII. The constraint fluid solver

There is a formulation of the linear approximation of (13) which can be solved using a direct sparse solver [46] in almost linear time but for the present application, it suffices to use Gauss-Seidel iterations. This is done without building the Schur matrix S_ϵ, working only with Jacobian matrices. In general, the regularization Σ improves the condition number of the linear system. When boundary and contact constraints are included, the system (16) becomes a mixed linear complementarity system.

It is customary in Gauss-Seidel iterations to directly update the approximate final velocity

$$\mathbf{v}_{k+1} = \mathbf{v}_k + \Delta t \mathbf{M}^{-1}\mathbf{f}_k + \mathbf{M}^{-1}\mathbf{G}^T\boldsymbol{\lambda}, \tag{30}$$

as the iterations proceed to improve the approximation of $\boldsymbol{\lambda}$. Because of this, it is more practical to introduce the unconstrained velocity as

$$\mathbf{v}_{k+1}^{(0)} = \mathbf{v}_k + \Delta t \mathbf{M}^{-1}\mathbf{f}_k, \tag{31}$$

and the final velocity estimate is then

$$\mathbf{v}_{k+1} = \mathbf{v}_{k+1}^{(0)} + \mathbf{M}^{-1}\mathbf{G}^T\boldsymbol{\lambda}, \tag{32}$$

once we know $\boldsymbol{\lambda}$. As the iterations progress, with $\nu = 1, 2, \ldots$, we update $\mathbf{v}_{k+1}^{(\nu)}$ as well as $\boldsymbol{\lambda}^{(\nu)}$ and we can monitor the constraint equation residual error as

$$\mathbf{r}^{(\nu)} = \mathbf{G}_k \mathbf{v}_{k+1}^{(\nu)} + \boldsymbol{\Sigma}\boldsymbol{\lambda}^{(\nu)} - \mathbf{c}, \tag{33}$$

where $\mathbf{c}$ is defined as in Eqn. (18). This is what is used in the algorithm VII.1 below.

A. *Gauss-Seidel solver for* SPOOK

The projected Gauss-Seidel (GS) method iterates through the individual constraint equations one by one, updating the current approximation of the Lagrange multipliers, based on the local residual error. GS is a linear relaxation method, with slow (linear) global convergence. However, it can be fast in the first few iterations and has the advantage that it smoothes the solutions thus damping out jitter and instabilities. This makes it useful for computer graphics and interactive simulations. It is also relatively simple to code and leaves a small memory footprint.

We can write the projected GS algorithm for our MLCP as follows.

Algorithm VII.1 Gauss-Seidel iterations to solve $(\mathbf{G}\mathbf{M}^{-1}\mathbf{G}^T + \boldsymbol{\Sigma})\boldsymbol{\lambda} = \mathbf{c} - \mathbf{G}\mathbf{v}_k$

```
Given $\mathbf{c}, \mathbf{M}, \mathbf{G}, \boldsymbol{\Sigma}, \Delta t, \mathbf{f}_k, \boldsymbol{\lambda}_0$ and $\mathbf{v}_k$.
Initialize $\mathbf{v} = \mathbf{v}_k + \Delta t \mathbf{M}^{-1}\mathbf{f}_k, \quad \boldsymbol{\lambda} = \boldsymbol{\lambda}_0$.
Compute initial residual and error, $\mathbf{r} = \mathbf{G}\mathbf{v} + \boldsymbol{\Sigma}\boldsymbol{\lambda} - \mathbf{c}$
Compute blocks $\mathbf{D}_{kk} = \sum_b \mathbf{G}_{kb}\mathbf{M}_{bb}^{-1}\mathbf{G}_{kb}^T$, for $k = 1, 2, \ldots, n_c$
repeat
    for $k = 1, 2, \ldots, m$ do                          ▷ Loop over constraints
        $\mathbf{r} = -\mathbf{c}_k + \boldsymbol{\Sigma}_{kk}\boldsymbol{\lambda}_k$                          ▷ Local residual
        for $b = b_{k_1}, b_{k_2}, \ldots, b_{n_{b_k}}$ do         ▷ Loop over bodies
            $\mathbf{r} = \mathbf{r} + \mathbf{G}_{kb}\mathbf{v}_b$
        end for
        Solve LCP: $0 \leq \mathbf{D}_{kk}\mathbf{z} + \mathbf{r}_k - \mathbf{D}_{kk}\boldsymbol{\lambda}_k \perp \mathbf{z} \geq 0$
        $\Delta\boldsymbol{\lambda}_k = \mathbf{z} - \boldsymbol{\lambda}_k$
        $\boldsymbol{\lambda}_k = \mathbf{z}$
        for $b = b_{k_1}, b_{k_2}, \ldots, b_{n_{b_k}}$ do         ▷ Loop over bodies
            $\mathbf{v}_b = \mathbf{v}_b + \mathbf{M}_{bb}^{-1}\mathbf{G}_{kb}^T\Delta\boldsymbol{\lambda}_k$        ▷ Update velocities
        end for
    end for
until Error is small, or iteration time is exceeded
```

Solving large MLCPs can be very difficult, but we only need to solve small ones. In the context of SPH, there is no friction even, and thus, we have one dimensional complementarity problems of the form

$$v = d \cdot z + q \geq 0, \qquad z \geq 0, \text{ and } wz = 0, \tag{34}$$

where $d > 0$. In other words, either the contact force z is positive and produces a zero velocity v, or the contact force vanishes and the velocity is positive. The solution to this is $z = -q/d$ when $q < 0$, and $z = 0$ otherwise.

B. Integration with rigid multibody mechanics

In our 3D examples, the non penetration frictionless constraints used for the fluid boundaries in section VI are very similar to that of two contacting rigid bodies. Thus, the constraint fluid model is easily integrated with rigid multibody mechanics. The net effect of non penetration constraints produces buoyancy of the rigid bodies fully consistent with Archimedes's principle. Consequently rigid bodies will automatically sink or float in the fluid depending on their given density and geometry. Also in our 2D cases with boundary fields, buoyancy falls out naturally and integration with constraint mechanics is consistent. We could incorporate other types of constraint modeling such as hinges and joints, as well as cloth type of constraints, or even deformable materials, for a very general and consistent multiphysics method, but this is beyond the scope of this paper.

VIII. Overall method description

The overall algorithm takes the form of Alg. (VIII.1).

Algorithm VIII.1 Overall algorithm for constraint fluid animation

```
repeat
    Detect particle neighbors within $h$
    Detect boundary condition violations
    Compute particle densities: Eqn. (2)
    Build Jacobians: Eqns. (23),(24),(29).
    Solve the MLCP, Alg. (VII.1)
    Update positions
    Visualize
until Simulation done
```

The broad phase neighbor search algorithm we use here is the sweep-and-prune method [49], but a variety of other methods could be used equivalently. Our 3D-simulations are implemented in AgX Multiphysics [50], and rendered in PovRay [51] with the fluid represented either by blob iso-surfaces, or by a marching cubes generated surface mesh. The 2D-simulations where implemented in Algodoo [52] and visualized using a GPU based meta surface algorithm.

IX. Results

A. Simulation of water in 3D

To illustrate near-incompressibility and proof of concept of the constraint fluid method, we fill a container of base area 0.36×0.36 m^2 with fluid as illustrated in Fig 1. The final scene consists of $10,000$ particles with 20 neighbors per particle on average. The total water volume is 70 liter corresponding to a mass per particle is set to 0.007 kg. The smoothing distance used is 0.03 m. The time step is 1/120 s, and the constraint relaxation time τ is four time steps. The resulting fluid volume is approximately 60 liters corresponding to an average density of 1170 kg/m^3 compared to a reference density of 1000 kg/m^3 for water. This is a 17% error. This compression comes from both the model and the numerical approximations. First, the regularization parameter $\epsilon = 10^{-3}$ relaxes the density constraint and is linearly related to the observed compressibility. The time step introduces numerical compressibility of order $O(h^{-2})$. The relatively low number of Gauss-Seidel iterations, 15 in our examples, also introduces errors. Least but not last is the fact that the kernel normalization criterion, Eqn. (3) is not satisfied near the boundaries due to particle deficiencies. Because of this deficiency, the density constraint is enforcing a higher density than what is necessary near the boundaries.

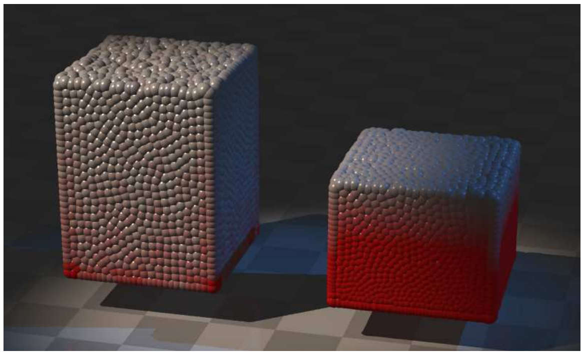

Fig. 1: *Color coded density profiles of the constraint fluid (left) compared with a standard* SPH *fluid (right), given the same size of the time step.*

As shown in Fig. 2, the density of the constraint fluid decreases at the very bottom of the container due to the deficiency, and goes up just above the bottom, to compensate for the deficiency. The situation would have been substantially improved if we had implemented the SPH boundary density field of our 2D-simulations also in the 3D-simulations. Also note that the deficiency neither depends on the height of the fluid pillar nor the local pressure.

To compare the constraint fluid with a standard SPH fluid we use the same number of particles, the same values for the time step, the particle mass and the smoothing length. In both cases, we use the poly-6 kernel function [12]. For the SPH specific parameters we use a sound speed of 1.58 ms^{-1}, and a viscosity of 0.5 kg$m^{-1}s^{-1}$. Given the time step and the number of particles in the fluid pillar, we didn't have much choice in choosing these two parameters, but had to set them to values that give a reasonably stable and still not too viscous fluid. Stability is a serious problem in standard SPH at these large time steps, and indeed our simulation is only barely stable and the resulting fluid is rather viscous, not much like water. The effective fluid volume is roughly 32 litres, corresponding to an average density of approximately 2190 kg/m^3, linearly increasing with the depth with extreme values at the bottom. The average number of neighbors per particle increases to nearly 35 and again, the variations increase with depth. The color coded density profiles of the constraint fluid and the standard SPH fluid, respectively, are visualized in Fig. 2. Red color indicates that the density is too high. Obviously standard SPH cannot represent the desired volume of fluid unless we would add several times as many particles, resulting in even more compression at the bottom and substantial stability problems.

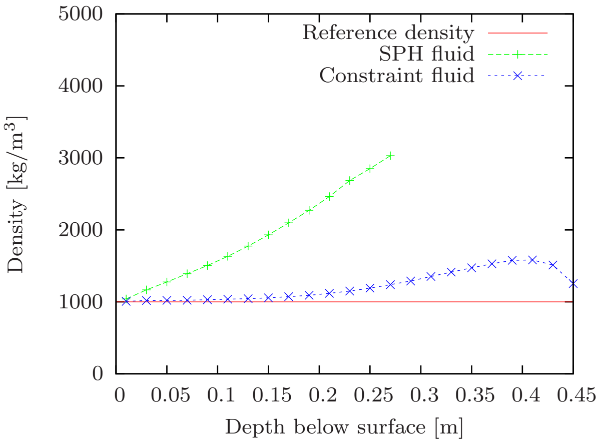

Fig. 2: *The mass density as a function of depth from the surface for standard* SPH *vs., the constraint fluid.*

Fig. 3 shows a sequence from the flushing of water into a container of bottom area 1.2 × 1.2 m^2, with other parameters kept as in previous examples, but with up to 250,000 particles. As is demonstrated in the supplementary video, the numerical viscosity of the fluid is relatively low even at these large time steps and few solver iterations, resulting in a vivid and realistic simulation of water.

The convergence rate of the iterative Gauss-Seidel procedure for a fluid pillar consisting of 100,000 particles is shown in Fig. 4 where the residual error of the system solved in Algorithm VII.1 is plotted against the number of Gauss-Seidel iterations. A relatively small 1% error is reached at 100 iterations. However, for interactive simulations and computer graphics applications we find that it is usually enough with 5 to 15 iterations for this system size. A relevant reflection is that a single Jacobi iteration corresponds to a standard pairwise SPH pressure penalty force of Eqn. (8), and additional iterations will propagate information further in the system. With Gauss-Seidel iterations, information propagates through the system already in the first iteration. This gives an intuitive picture of the implicit nature of our pressure model, and an explanation of why already a few iterations give dramatic stability improvements. However, the Gauss-Seidel propagation direction is non-deterministic, and the ordering of the Gauss-Seidel iterations is typically random, and therefore this process does not have an obvious physical meaning. One should therefore refrain from interpreting this as a physical impulse-propagation method, as often is done in the literature on physics based animation.

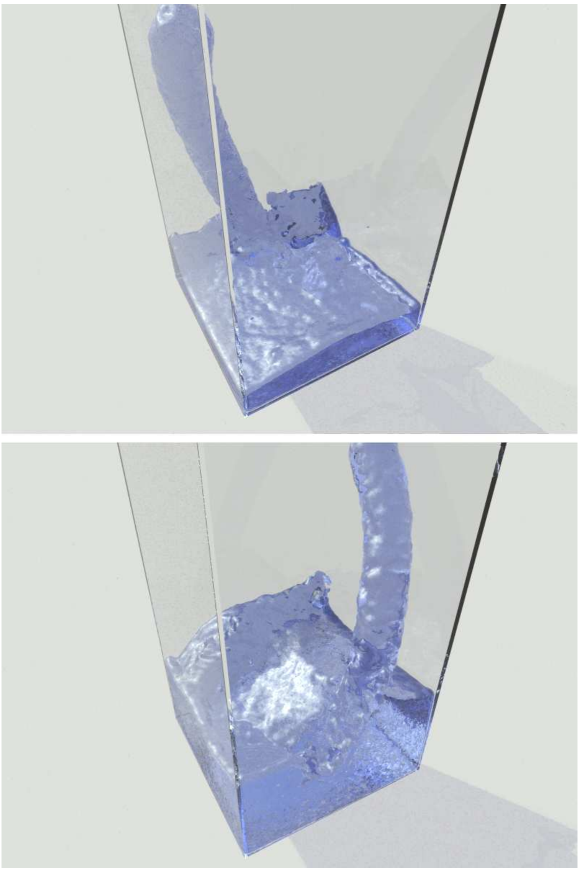

Fig. 3: *Water flushing into a container.*

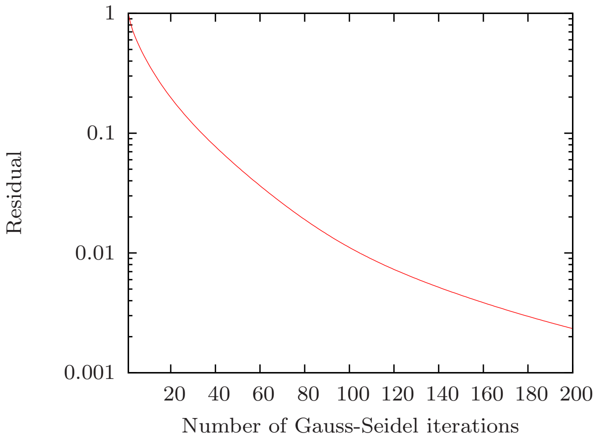

Fig. 4: *Convergence of the Gauss-Seidel iterations for a 100,000 particle fluid pillar.*

In Fig. 5 we illustrate how buoyancy falls out automatically from the boundary condition model. The rigid bodies are affected by gravity, as well as pressure forces from the fluid particles. Dense bodies sink and pile up at the

Fig. 5: *Water in a container, with dense bodies sinking and light bodies forming a floating pile on the surface.*

bottom, and bodies less dense than water float and form a pile on the surface. The entire system is solved as a clean and consistent multiphysics system.

B. Two dimensional examples

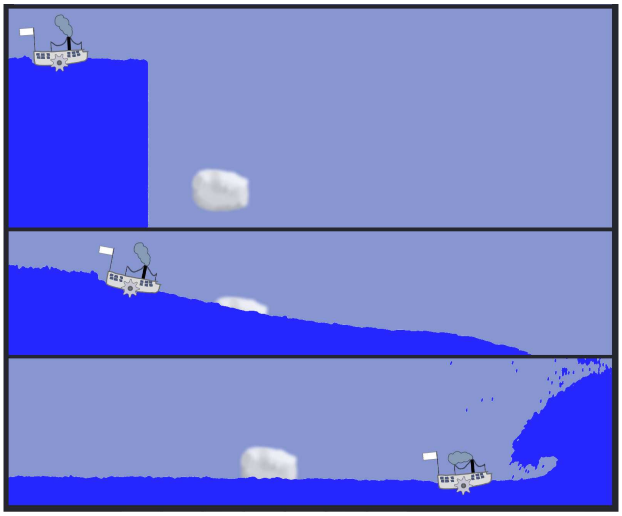

Fig. 6: *The 2D dam break scenario is a standard benchmark in fluid simulation. A troubled steam paddler was added to the water pillar to help visualizing the flow in a dramatic way.*

Dam break simulations are often used as benchmark tests in computational fluid dynamics [12]. The sequence with the troubled steam paddler in Fig. 6 and the supplementary video, shows the characteristic behavior of the propagating wave front and the reflected wave. The simulation is using 12600 particles. As described previously, our method allows us to model the steam paddler consisting of 35 jointed rigid bodies and a motor using constrained rigid body dynamics, and to solve for both the fluid and the paddler dynamics with one single solve.

A pump is an obvious example of a system that requires a near incompressible fluid model. To further illustrate this and the integration with constrained rigid body systems, an idealized but fully functional piston pump with valves is shown in Fig. 7. The simulation consists of 11500 particles and 6 jointed rigid bodies. Numerous hydro mechanical devices and phenomena could be modeled in this way.

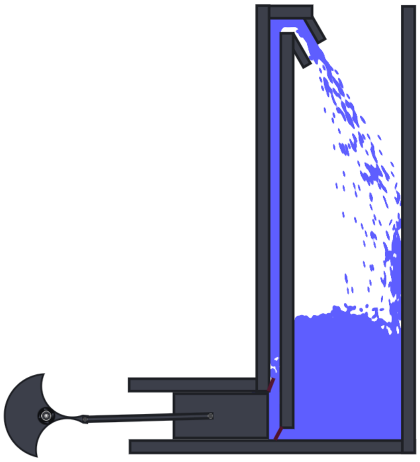

Fig. 7: *A simple 2D piston pump with valves.*

C. Performance

The computational time of both standard SPH and the constraint fluid is dominated by the following: a) finding neighbors; b) calculating the SPH density; c) calculating the pressure forces. The neighbor search time is in principle the same in both models, and under normal circumstances this is about 40% of the computational cost for one time step in our implementation. However, in many cases the compression artifacts in standard SPH give a dramatic increase in the number of neighbors, in particular when simulating high water pillars subject to gravity and other loads exerting pressure. Likewise, the time for computing the SPH density is similar in both methods and typically takes 20% of the time in standard SPH, but also scales up in cost with the number of neighbors due to compression artifacts. The force computations in standard SPH, use the remaining 40% of the compute time, but also scale up with the number of neighbors. When solving for the geometric pressure forces in the constraint fluid, the time for a single Gauss-Seidel iteration is comparable to the time for the force computation in standard SPH and scales linearly with the number of iterations. We typically use 5 to 15 iterations, and thus 40% of the computations are 5 to 10 times slower than in standard SPH in principle. But when compression is taken into account, the constraint fluid is as fast or faster than standard SPH, with the difference that the constraint fluid has much lower viscosity, is nearly incompressible and results in a much more stable, and realistic simulation. Tuning standard Sph to this level of fidelity requires a 100-1000 times smaller time step, making it very slow and essentially impractical for simulating large volumes of water. Thus, for 10 iterations, the constraint fluid is between 25 to 250 times more efficient than standard Sph.

The 3D-simulations were run on an Intel 2.8 GHz Xeon processor allowing for 6000 constraint fluid particles at 10 Hz using serial and non-optimized code, with linear scaling to larger systems. 2D-simulations were run on an Intel 2.53 GHz laptop, with serial code, allowing for 12000 particles at 10 Hz, also with linear scaling to larger systems.

X. Conclusion and future work

The constraint fluid method presented above can be improved using the same schemes developed for the standard Sph method [12], including the use of adaptive and asymmetric kernel functions, and density renormalization, for instance. For large simulations it would also be desirable to apply adaptive sampling of the particle resolution [53], [54]. The constraint fluid has low numerical viscosity and viscosity models can be added either using the traditional force based viscosity terms of Sph, Eqn. (9), as external forces in Spook, Eqns. (18); by increasing the constraint damping time, τ, in Eqn. (19); or by constructing a viscosity constraint using Rayleigh dissipation functions in the discrete variational framework of Spook [35]. The latter is recommended, since this allows for very high viscosity also at large time steps. The MLCP solve stage can be improved using a parallel implementation of the preconditioned conjugate gradient method, for instance, and this can be implemented on stream and multicore processors with substantial speedup, promising interactive frame rates for several hundred thousand particles. The iterations can also be warm started using the solutions of the previous time step, thus exploiting the temporal coherence of the system.

In conclusion, we have presented a novel fluid simulation method with the following advantages: a) it can efficiently handle a high degree of incompressibility; b) it is stable for large time steps; c) the constraint model has a physics based origin and is fully consistent with models of rigid body mechanics and multiphysics systems; d) both computationally expensive stages, namely, collision detection and the linear solve, can each be parallelized.

It is our belief that constraint fluids can become competitive in computer graphics, with applications ranging from animation and visual effects to real-time simulation in simulators and computer games. However, the scope of the method is broader, and constraint fluids should also be useful in the domain of education, engineering and basic science, and offer rich opportunities for further development and research.

Acknowledgment

This research was conducted using resources of High Performance Computing Center North (HPC2N), Umeå university, and supported in part by the Swedish Foundation for Strategic Research (SSF) under the frame program grant SSF-A3 02:128; and by Vinnova/SSF grant VINST #247; by the Kempe foundations, grant JCK-2613; and by Vinnova/ProcessIT Innovations. The authors wish to acknowledge Bo Kågström and Krister Wiklund for valuable discussions, and Emil Ernerfeldt, Nils Hjelte, Niklas Melin, Martin Nilsson, Fredrik Nordfeldth and other staff at Algoryx for help with programming and rendering.

References

[1] M. Desbrun and M. P. Cani, "Smoothed particles: A new paradigm for animating highly deformable bodies," in *Computer Animation and Simulation '96 (Proceedings of EG Workshop on Animation and Simulation)*. Springer-Verlag, 1996, pp. 61–76.

[2] D. Stora, P.-O. Agliati, M.-P. Cani, F. Neyret, and J.-D. Gascuel, "Animating lava flows," in *Graphics Interface (GI'99) Proceedings*, Jun 1999, pp. 203–210. [Online]. Available: http://artis.imag.fr/Publications/1999/SACNG99

[3] M. Müller, D. Charypar, and M. Gross, "Particle-based fluid simulation for interactive applications," in *SCA '03: Proceedings of the 2003 ACM SIGGRAPH/Eurographics symposium on Computer animation*, 2003, pp. 154–159.

[4] M. Müller, S. Schirm, and M. Teschner, "Interactive blood simulation for virtual surgery based on smoothed particle hydrodynamics," *Journal of Technology and Health Care*, 2003.

[5] W. Liu, C. Sewell, N. Blevins, K. Salisbury, K. Bodin, and N. Hjelte, "Representing fluid with smoothed particle hydrodynamics in a cranial base simulator," in *Proceedings of Medicine Meets Virtual Reality (MMVR), Long Beach, CA*, 2008, pp. 257–259.

[6] M. Müller, B. Solenthaler, R. Keiser, and M. Gross, "Particle-based fluid-fluid interaction," in *Eurographics/ACM SIGGRAPH Symposium on Computer Animation 2005*, 2005, pp. 1–7.

[7] T. Lenaerts and P. Dutré, "Mixing fluids and granular materials," in *Proceedings of Eurograpics 2009*, P. Dutré and M. Stamminger, Eds. Eurographics, 2009.

[8] M. Müller, R. Keiser, A. Nealen, M. Pauly, M. Gross, and M. Alexa, "Point based animation of elastic, plastic and melting objects," in *Proceedings of the ACM SIGGRAPH/EUROGRAPHICS Symposium on Computer Animation*, Aug 2004.

[9] L. B. Lucy, "A numerical approach to the testing of the fission hypothesis," *The Astronomical Journal*, vol. 82, pp. 1013–1024, 1977.

[10] R. A. Gingold and J. J. Monaghan, "Smoothed particle hydrodynamics: theory and application to non-spherical stars," *Monthly Notices of the Royal Astronomical Society*, vol. 181, pp. 375–398, 1977.

[11] S. Li and W. K. Liu, "Meshfree particle methods and their applications," *Appl. Mech. Rev.*, vol. 55, Jan. 2002.

[12] G. R. G.-R. Liu and M. B. Liu, *Smoothed particle hydrodynamics: a meshfree particle method*. Singapore: World Scientific, 2003.

[13] J. J. Monaghan, "Smoothed particle hydrodynamics," *Reports on Progress in Physics*, vol. 68, no. 8, pp. 1703–1759, 2005. [Online]. Available: http://stacks.iop.org/0034-4885/68/1703

[14] S. Rusinkiewicz and M. Levoy, "Qsplat: A multiresolution point rendering system for large meshes," in *Proceedings of ACM SIGGRAPH '00*, 2000.

[15] M. Zwicker, M. Pauly, O. Knoll, and M. Gross, "Pointshop 3d: An interactive system for point-based surface editing," in *Proceedings of ACM SIGGRAPH '02*, 2002.

[16] M. Levoy, "Display of surfaces from volume data," *EEE Computer Graphics and Applications*, vol. 8, pp. 29–37, May 1988.

[17] W. McNeely, K. Puterbaugh, and J. Troy, "Six degree-of-freedom haptic rendering using voxel sampling," in *Proceedings of ACM SIGGRAPH '99*, August 1999, pp. 401–408.

[18] K. L. Palmerius, M. Cooper, and A. Ynnerman, "Haptic rendering of dynamic volumetric data," *IEEE Transactions on Visualization and Computer Graphics*, vol. 14, no. 2, pp. 263–276, March 2008.

[19] W. T. Reeves, "Particle system-a technique for modeling a class of fuzzy objects," in *Proceedings of ACM SIGGRAPH '83*, vol. 17, July 1983, pp. 359–376.

[20] M. Levoy and T. Whitted, "The use of points as a display primitive," Technical Report 85-022, Computer Science Department, January 1985.

[21] H. Pfister, M. Zwicker, J. van Baar, and M. Gross, "Surfels: surface elements as rendering primitives," in *Proceedings of ACM SIGGRAPH '00*, 2000.

[22] M. Gross and H. Pfister, *Point-Based Graphics*. Morgan-Kauffman, July 2007.

[23] K. F. R. Courant and H. Lewy, "On the partial difference equations of mathematical physics, english translation of the 1928 german original," *IBM Journal*, pp. 215–234, 1967.

[24] R. Bridson, *Fluid Simulation for Computer Graphics*. A.K. Peters, 2008.

[25] S. J. Cummins and M. Rudman, "An SPH projection method," *Journal of Computational Physics*, vol. 152, pp. 584–607, 1999.

[26] S. Koshizuka and Y. Oka, "Moving-particle semi-implicit method for fragmentation of incompressible fluid," *Nuclear Science Engineering*, vol. 123, pp. 421–434, Jul. 1996.

[27] S. Premoze, T. Tasdizen, J. Bigler, A. Lefohn, and R. T. Whitaker, "Particlebased simulation of fluids," in *Proceedings of Eurographics '03*, vol. 22, no. 3, 2003, pp. 401–410.

[28] E.-S. Lee, C. Moulinec, R. Xuc, D. Violeau, D. Laurence, and P. Stansby, "Comparisons of weakly compressible and truly incompressible algorithms for the SPH mesh free particle method," *J. Comp. Phys.*, vol. 227, pp. 8417–8436, 2008.

[29] M. Ellero, M. Serrano, and P. Español, "Incompressible smoothed particle hydrodynamics," *Journal of Computational Physics*, vol. 226, no. 2, pp. 1731 – 1752, 2007. [Online]. Available: http://www.sciencedirect.com/science/article/B6WHY-4P2YWYC-5/2/611488181b994c812ccd286bcef63457

[30] E. Hairer, C. Lubich, and G. Wanner, *Geometric Numerical Integration*, ser. Spring Series in Computational Mathematics. Berlin, Heidelberg, New York, London, Paris, Tokyo, Hong Kong: Springer-Verlag, 2001, vol. 31.

[31] B. Solenthaler and R. Pajarola, "Predictive-corrective incompressible SPH," in *Proceedings of ACM SIGGRAPH '09*, 2009.

[32] M. Müller, B. Heidelberger, M. Hennix, and J. Ratcliff, "Position based dynamics," *J. Vis. Comun. Image Represent.*, vol. 18, no. 2, pp. 109–118, 2007.

[33] G. Batchelor, *An Introduction to Fluid Dynamics*. Cambridge University Press, 1967.

[34] M. Becker and M. Teschner, "Weakly compressible SPH for free surface flows," in *Proc. ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, 2007, pp. 63–72.

[35] C. Lacoursière, "Ghosts and machines: Regularized variational methods for interactive simulations of multibodies with dry frictional contacts," Ph.D. dissertation, Department of Computing Science, Umeå University, Sweden, Jun. 2007. [Online]. Available: http://urn.kb.se/resolve?urn=urn:nbn:se:umu:diva-1143

[36] U. Ascher, H. Chin, L. Petzold, and S. Reich, "Stabilization of constrained mechanical systems with DAEs and invariant manifolds," vol. 23, pp. 135–158, 1995.

[37] F. A. Bornemann, *Homogenization in Time of Singularly Perturbed Mechanical Systems*, ser. Lecture notes in mathematics, 0075-8434. Berlin: Springer, 1998, vol. 1687.

[38] H. Rubin and P. Ungar, "Motion under a strong constraining force," *Communications on Pure and Applied Mathematics*, vol. X, pp. 65–87, 1957.

[39] E. Hairer and G. Wanner, *Solving Ordinary Differential Equations II: Stiff and Differential Algebraic Problems*, 2nd ed., ser. Springer Series in Computational Mathematics. Berlin, Heidelberg, New York, London, Paris, Tokyo, Hong Kong: Springer-Verlag, 1996, vol. 14.

[40] J. Baumgarte, "Stabilization of constraints and integrals of motion in dynamical systems," vol. 1, no. 1, pp. 1–16, 1972.

[41] C. Lanczos, *The Variational Pinciples of Mechanics*, 4th ed. New York: Dover Publications, 1986.

[42] V. I. Arnold, *Mathematical Methods of Classical Mechanics*, 2nd ed., ser. Graduate Texts in Mathematics. New York: Springer-Verlag, 1989, vol. 60, translated from the Russian by K. Vogtmann and A. Weinstein.

[43] A. J. Kurdila and F. J. Narcowich, "Sufficient conditions for penalty formulation methods in analytical dynamics," *Computational Mechanics*, vol. 12, pp. 81–96, 1993.

[44] M. Servin and C. Lacoursière, "Rigid body cable for virtual environments," *IEEE Transactions on Visualization*, vol. 14, no. 4, pp. 783–796, 2008.

[45] M. Servin, C. Lacoursière, and N. Melin, "Interactive simulation of elastic deformable materials," in *Proc. of SIGRAD Conference 2006*, 2006, pp. 22–32.

[46] C. Lacoursière, "Regularized, stabilized, variational methods for multibodies," in *The 48th Scandinavian Conference on Simulation and Modeling (SIMS 2007)*, D. F. Peter Bunus and C. Führer, Eds. Linköping University Electronic Press, October 2007, pp. 40–48.

[47] C. T. Kelley, *Iterative Methods for Linear and Nonlinear Equations*, ser. SIAM Frontiers, 1995, vol. 16.

[48] M. Yildiz, R. A. Rook, and A. Suleman, "SPH with the multiple boundary tangent method," *International Journal for Numerical Methods in Engineering*, vol. 77, pp. 1416–1438, 2009. [Online]. Available: http://dx.doi.org/10.1002/nme.2458

[49] B. V. Mirtich, "Impulse-based dynamic simulation of rigid body systems," Ph.D. dissertation, University of California at Berkeley, Berkeley, CA, USA, 1996.

[50] AgX, "Agx multiphysics sdk," http://www.algoryx.se/agx, 2009.

[51] POV-Ray, "Persistence of vision raytracer. v.3.6," http://www.povray.org, 2008.

[52] Algodoo, "Algodoo 2d physics sandbox v5.42," http://www.algoryx.se/algodoo, 2009.

[53] S. Kitsionas and A. P. Whitworth, "Smoothed particle hydrodynamics with particle splitting, applied to self-gravitating collapse," *Monthly Notices of the Royal Astronomical Society*, vol. 330, pp. 129–136, 2002.

[54] B. Adams, M. Pauly, R. Keiser, and L. J. Guibas, “Adaptively sampled particle fluids,” in *ACM Transactions on Graphics (SIGGRAPH ’07 papers)*, vol. 26, no. 3. New York, NY, USA: ACM Press, 2007, p. 48.

[55] M. Müller, B. Heidelberger, M. Hennix, and J. Ratcliff, “Position based dynamics,” in *Proc. 3rd workshop in Virtual Reality Interactions and Physical Simulation*, 2006.

[56] L. Kharevych, W. Yang, Y. Tong, E. Kanso, J. E. Marsden, P. Schröder, and M. Desbrun, “Geometric variational integrators for computer animation,” in *SCA ’06: Proceedings of the 2006 ACM SIGGRAPH/Eurographics symposium on Computer animation*. Aire-la-Ville, Switzerland, Switzerland: Eurographics Association, 2006, pp. 43–51.

[57] H. C. Elman, D. J. Silvester, and A. J. Wathen, *Finite Elements and Fast Iterative Solvers: with applications in incompressible fluid dynamics*. Oxford University Press, 2005.

[58] S. Koshizuka, Y. Oka, and H. Tamako, “A particle method for calculating splashing of incompressible viscous fluid,” in *Proceedings of the International Conference on Mathematics and Computations, Reactor Physics and Environmental Analyses*, vol. 2. ANS, 1995, pp. 1514–.

[59] F. Losasso, T. Shinar, A. Selle, and R. Fedkiw, “Multiple interacting liquids,” in *Proceedings of ACM SIGGRAPH ’06*, vol. 25, 2006, pp. 812–819.

[60] M. Carlson, P. J. Mucha, R. B. Van Horn, III, and G. Turk, “Melting and flowing,” in *SCA ’02: Proceedings of the 2002 ACM SIGGRAPH/Eurographics symposium on Computer animation*. New York, NY, USA: ACM, 2002, pp. 167–174.

[61] Y. Zhu and R. Bridson, “Animating sand as a fluid,” in *Proceedings of ACM SIGGRAPH ’05*, 2005.

[62] F. Colin, R. Egli, and F. Lin, “Computing a null divergence velocity field using smoothed particle hydrodynamics,” *Journal of Computational Physics*, vol. 217, pp. 680–692, 2006.

[63] W. G. Hoover, *Smooth Particle Applied Mechanics: The State of the Art*. World Scientific, 2006.

[64] M. Kass and G. Miller, “Rapid, stable fluid dynamics for computer graphics,” in *Proceedings of ACM SIGGRAPH ’90*, vol. 24, 1990, pp. 49–57.

[65] J. Stam, “Stable fluids,” in *Proceedings of ACM Siggraph ’99*, 1999, pp. 121–128. [Online]. Available: citeseer.ist.psu.edu/stam99stable.html

[66] J. Chen and N. Lobo, “Toward interactive-rate simulation of fluids with moving obstacle using the navier-stokes equations,” *Computer Graphics and Image Processing*, vol. 57, pp. 107–116, 1994.

[67] N. Foster and D. Metaxas, “Realistic animation of liquids,” *Graph. Models and Image Processing*, vol. 58, 1996.

[68] N. Foster and R. Fedkiw, “Practical animation of liquids,” in *Proceedings of ACM SIGGRAPH ’01*, 2001, pp. 23–30.

[69] D. Enright, R. Fedkiw, J. Ferziger, and I. Mitchell, “Animation and rendering of complex water surfaces,” *ACM Transactions on graphics*, vol. 21, no. 3, pp. 736–744, 2002.

[70] M. R. Hestenes and E. Stiefel, “Methods of conjugate gradients for solving linear systems,” *J. Research Nat. Bur. Standards*, vol. 49, pp. 409–436, 1952.

[71] F. Losasso, J. O. Talton, N. Kwatra, and R. Fedkiw, “Two-way coupled SPH and particle level set fluid simulation,” *IEEE Trans. on vis. and computer graphics*, vol. 14, no. 4, pp. 797–804, jul–aug 2008.

[72] G. R. G.-R. Liu and M. B. Liu, *Smoothed particle hydrodynamics: a meshfree particle method*. Singapore: World Scientific, 2003.

[73] K. Lundin, M. Sillen, M. Cooper, and A. Ynnerman, “Haptic visualization of computational fluid dynamics data using reactive forces,” in *Visualization and Data Analysis 2005*, Erbacher, RF and Roberts, JC and Grohn, MT and Borner, K, Ed., vol. 5669. Spie-Int Soc Optical Engineering, 2005, pp. 31–41.

[74] M. Anitescu, “Optimization-based simulation of nonsmooth rigid multibody dynamics,” *Math. Program.*, vol. 105, no. 1, Ser. A, pp. 113–143, 2006.

[75] R. Goldenthal, D. Harmon, R. Fattal, M. Bercovier, and E. Grinspun, “Efficient simulation of inextensible cloth,” in *SIGGRAPH ’07*. New York, NY, USA: ACM Press, 2007, p. 49.

Kenneth Bodin Kenneth Bodin is a researcher at the High Performance Computing Center North at Umeå University and CEO at Algoryx Simulations. He completed his MSc at the physics department of Umeå University in 1989 and his PhLic in Theoretical Physics from Chalmers University of Technology in 1992. His research interests include computational physics, high performance computing, computer visualization and condensed matter physics.

Claude Lacoursière Claude Lacoursière is a researcher at the High Performance Computing Center North (HPC2N) and the UMIT center for industrial and IT research at Umeå University. He completed his MSc at the physics department of McGill University in 1993 and his PhD at Umeå University in 2007. He has 12 years of experience from industry research and development of multibody system simulations. His research interests include physics motivated numerical methods for real-time integration of mechanical systems, especially contacting multibodies subject to dry friction.

Martin Servin Martin Servin is senior lecturer at the department of Physics at Umeå University. He received his MSc and PhD degrees in theoretical physics from Umeå University in 1999 and 2003, respectively. His research interests include physical modeling and numerical simulation of complex mechanical systems with particular interest for real-time interactive 3D graphics and applications to robotics and off-road vehicles.