Real-time Height-field Simulation of Sand and Water Mixtures

Haozhe Su
haozhesu@global.tencent.com
LightSpeed Studios
Los Angeles, CA, USA

Siyu Zhang
cuzhang@global.tencent.com
LightSpeed Studios
Los Angeles, CA, USA

Zherong Pan
zrpan@global.tencent.com
LightSpeed Studios
Seattle, WA, USA

Mridul Aanjaneya
ma635@cs.rutgers.edu
Rutgers University
New Brunswick, NJ, USA

Xifeng Gao
xifgao@global.tencent.com
LightSpeed Studios
Seattle, WA, USA

Kui Wu
kwwu@global.tencent.com
LightSpeed Studios
Los Angeles, CA, USA

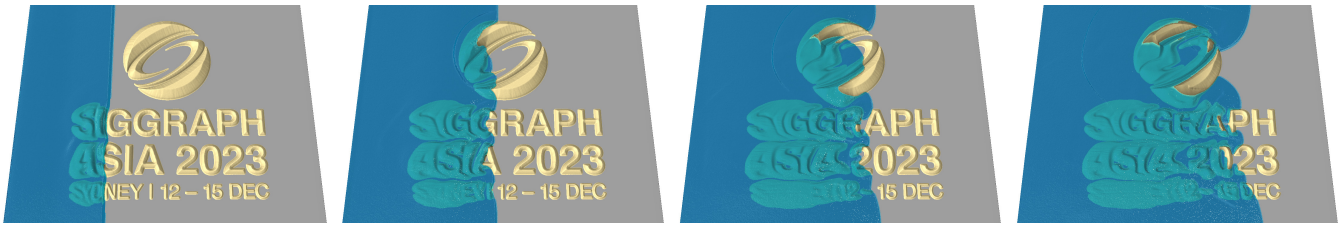


Figure 1: ***Flush:*** *The sand letters and logo are washed away by the flow.*

ABSTRACT

We propose a height-field-based real-time simulation method for sand and water mixtures. Inspired by the shallow-water assumption, our approach extends the governing equations to handle two-phase flows of sand and water using height fields. Our depth-integrated governing equations can model the elastoplastic behavior of sand, as well as sand-water-mixing phenomena such as friction, diffusion, saturation, and momentum exchange. We further propose an operator-splitting time integrator that is both GPU-friendly and stable under moderate time step sizes. We have evaluated our method on a set of benchmark scenarios involving large bodies of heterogeneous materials, where our GPU-based algorithm runs at real-time frame rates. Our method achieves a desirable trade-off between fidelity and performance, bringing an unprecedentedly immersive experience for real-time applications.

CCS CONCEPTS

• **Computing methodologies → Physical simulation**.

KEYWORDS

Sand and Water Mixture, 2.5D Simulation, Shallow Water Equations

ACM Reference Format:
Haozhe Su, Siyu Zhang, Zherong Pan, Mridul Aanjaneya, Xifeng Gao, and Kui Wu. 2023. Real-time Height-field Simulation of Sand and Water Mixtures. In *SIGGRAPH Asia 2023 Conference Papers (SA Conference Papers '23), December 12–15, 2023, Sydney, NSW, Australia.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3610548.3618159

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

SA Conference Papers '23, December 12–15, 2023, Sydney, NSW, Australia

© 2023 Copyright held by the owner/author(s). Publication rights licensed to ACM.

ACM ISBN 979-8-4007-0315-7/23/12…$15.00
https://doi.org/10.1145/3610548.3618159

1 INTRODUCTION

Water and sand are both ubiquitous in our daily life. Many people have likely enjoyed the experience of playing with sand at the beach, where damp sand retains its shape, allowing for the creation of intricate sandcastles, only to be washed away by the powerful forces of the waves. Replicating these phenomena is a challenging research topic. Although significant progress for offline applications [Gao et al. 2018a; Tampubolon et al. 2017] has been made over the past decade, we are still missing these phenomena in real-time applications, such as video games.

Several directions of research have contributed to the maturity of high-quality 3D simulation of sand and water mixtures. Widely used multi-physics simulation algorithms employ the meshless methods, which use particles to represent materials with different attributes or phases, such as Smoothed Particle Hydrodynamics (SPH) [Alduán and Otaduy 2011; Lenaerts and Dutré 2009; Ren et al. 2021; Yan et al. 2016; Yang et al. 2017], Discrete-Element Method (DEM) [Bell et al. 2005], and hybrid SPH-DEM [Bell et al. 2005; Rungjiratananon et al. 2008; Wang et al. 2021]. However, these methods become intractable when we need to handle billions of particle-particle interactions in large-scale 3D scenarios. To mitigate the computational cost, hybrid Eulerian/Lagrangian methods, such as the Material Point Method (MPM) [Daviet and Bertails-Descoubes 2016; Gao et al. 2018a; Klár et al. 2016; Tampubolon et al. 2017], are naturally more efficient due to the interactions between particles being handled on a coarse Eulerian grid, while the dynamics are traced in Lagrangian space using particles. Although significant efforts have been made to accelerate both SPH and MPM methods via GPU parallelization [Gao et al. 2018b; Goswami et al.

2010], hierarchical [Solenthaler and Gross 2011], or advanced numerical algorithms [Wang et al. 2020], dealing with large-scale scenarios still requires millions of particles, making them intractable for real-time applications.

Regarding real-time applications such as games, trade-offs have to be made between fidelity and performance in order to fit the computation into low-end desktop machines. There has been a body of research works on real-time fluid simulation techniques based on simplified governing equations, including reduced-order models [Treuille et al. 2006], Wave Equation [Jeschke and Wojtan 2017; Yuksel et al. 2007], and Shallow-Water Equations (SWE) [Chentanez and Müller 2010; Layton and van de Panne 2002; Thurey et al. 2007]. Recently, the shallow water assumption has been further applied to simulate dry granular flows efficiently [Zhu and Yang 2010]. A key component in all these techniques is dimension reduction. In particular, the (S)WE equations use 2.5D height-field-based data structures to reduce the number of decision variables by several orders of magnitude. However, all these works are focused on a homogeneous material model, and none of them can simulate the interaction between multiple phases, which, up-to-date, is only possible via full 3D modeling.

In this work, we propose the first height-field-based real-time framework to simulate sand, water, and their mixture. Inspired by 2.5D sand simulation [Zhu and Yang 2010], our method derives a depth-integrated 2.5D governing equation that allows the modeling of the well-studied elastoplastic behavior of sand, as well as sand-water-mixing phenomena, on top of the existing shallow water framework. We present a novel spatial-temporal discretization scheme for our governing equations and use operator-splitting to handle each force term, including water diffusion, external friction, internal elastoplastic force, and momentum exchange between sand and water. We introduce a piecewise linear function to define the relationship between sand cohesion and saturation. We further adopt an asynchronous update routine to improve the performance. Putting together, our method allows GPU-friendly simulation of large-scale sand-water scenarios at real-time frame rates. We demonstrate our approach with various sand-water coupling phenomena. We summarize our contributions as follows:

- A 2.5D governing equation for sand-water mixtures.
- A grid-based elastoplastic formulation for sand.
- A piecewise linear saturation control function.
- A semi-implicit operator splitting discretization scheme.

2 BACKGROUND

In this section, we briefly review previous works on sand simulation, sand-water mixture, and height-field simulation methods.

Particle-based sand simulation needs to model millions of individual grains, and the straightforward way is to model the sand as interactions between individual particles [Luciani et al. 1995]. Following this direction, researchers introduced several approaches to handle the interaction between sand particles, e.g., Discrete-Element Method (DEM) [Bell et al. 2005], Smoothed Particle Hydrodynamics (SPH) [Alduán and Otaduy 2011; Benes et al. 2006], hybrid SPH-DEM approach [Bell et al. 2005], and Projective Peridynamics [He et al. 2018]. Recently, Takahashi and Batty [2021] proposed a monolithic pressure-friction-contact solver to simulate rigid bodies' two-way interactions with continuum granular materials. The major drawback of particle-based methods is the high computational cost of detecting and handling particle-particle interactions, which can be intractable in large-scale 3D scenarios with billions of particles.

Hybrid Lagrangian/Eulerian sand simulation is naturally more efficient in handling the interactions between particles in Eulerian space while tracing the dynamics in Lagrangian space. Zhu and Bridson [2005] modified the Particle-in-Cell (PIC) method to simulate sand as an incompressible fluid with frictional stress, which is extended later with a unilateral variational constraint [Narain et al. 2010]. By allowing particles to carry more attributes such as deformation gradient, the Material Point Method (MPM) has been demonstrated to reproduce realistic visual effects for sand simulations [Daviet and Bertails-Descoubes 2016; Klár et al. 2016]. Yue et al. [2018] also introduced a hybrid method to efficiently couple a continuum material point simulation with a discrete element simulation. Unfortunately, even accelerated by the GPU [Gao et al. 2018b], the Hybrid Lagrangian/Eulerian method still cannot meet the requirements of real-time applications.

Eulerian-based simulation typically has a lower memory cost with higher performance than the particle-based method due to fewer degrees of freedom. It is a prevalent method for simulating materials in games, such as stable fluids [Stam 1999]. Koike et al. [2020] accelerated the Eulerian fluid simulation with an asynchronous time integrator. Levin et al. [2011] developed an Eulerian solid simulation method with contact, which was extended later to couple deformable objects with incompressible fluids [Teng et al. 2016]. Even using the grid, it takes minutes to solve the fluid and solid dynamics, which makes it impossible to deploy in games.

Sand-water mixture can be modeled by treating water and sand as particles of different types, phases, or materials. Several coupling frameworks have been proposed to model interactions between fluids and granular materials, e.g., SPH-DEM [Rungjiratananon et al. 2008] and unified SPH [Lenaerts and Dutré 2009; Ren et al. 2021; Yan et al. 2016; Yang et al. 2017]. Recently, Wang et al. [2021] also introduced a capillary model to capture various soil-structure destruction phenomena under the SPH-DEM framework. Tampubolon et al. [2017] used a two-grid MPM to discretize the water and sand continuum equations. Similarly, Gao et al. [2018a] solved the momentum exchange between fluid and sediment through two background grids. Our method borrows a similar idea to simulate water and sand using different background height fields and exchange the momentum between these two fields.

Height-field simulation is a popular approach to accelerating 3D simulation in real-time applications such as games. Benes and Forsbach [2001] introduced layered representation for terrain erosion simulation, which was later extended in [Šťava et al. 2008] to achieve interactive modeling. Fei et al. [2019] also proposed a shallow water-style approach to address non-Newtonian fluids along the hair strand. Sumner et al. [1999] deformed the ground surface to model sand, mud, and snow, and [Onoue and Nishita 2005] extended this idea by adding rigid body interactions for non-physical editing. Layton and van de Panne [2002] first introduced Shallow Water Equations (SWE) to the computer graphics community, which included a 2D velocity field in addition to the water

columns. Since waterfalls or overturning waves cannot be represented by a 2D height field, phenomena that can be simulated using SWE are limited. Thurey et al. [2007] resolved breaking waves by generating and evolving proxy patches, and [Chentanez and Müller 2010] introduced particles to the SWE system for waterfalls. Later, the 3D grid, height field, and particles were combined into one system for large-scale water with details [Chentanez et al. 2015]. Hagen et al. [2005] addressed complex irregular boundaries in SWE with a finite volume method. The height field simulation idea was also utilized by Zhu and Yang [2010] to simulate sand as a surface flow. Recently, Zhu et al. [2021] modeled dry granular flows by depth-integrating three-dimensional governing equations along its vertical direction, yielding the Shallow Sand Equations (SSE). This work extends SSE wtih an elastoplastic internal force and a coupling scheme for sand-water mixture.

3 PHYSICAL MODELS

In this section, we first introduce the layered height field for representing sand-water mixtures, then describe the governing equations for the two phases, and finally introduce the two-way coupling.

3.1 Layered Height-field Representation

Our representation is based on the shallow assumption [Randall 2006], i.e., the vertical scale along the z-axis is negligible as compared with the horizontal scale so that 3D governing equations can be depth-integrated to eliminate the vertical velocity. We use h and $\mathbf{v}$ to denote height and velocity fields, respectively. We also use $h(x, y)$ and $\mathbf{v}(x, y)$ to indicate the height and velocity at location $\mathbf{x} = (x, y)$, where x and y denote horizontal axis-wise components. For brevity, we ignore the parameters without confusion. *(See Table 1 for a complete notation list.)*

Our framework uses a discrete layered height-field model that unifies the representation of different materials, as illustrated in Fig. 2. The fluid can flow freely above the sand while its movement is impeded by the grains when inside the sand. Due to the significantly different dynamics between the water phases, the water column is divided into pure water and mixed water with sand. In terms of sand dynamics, we simulate the entire sand column without considering the vertical variations and only count the external kinetic friction against the bottom of the sand and the top of the terrain. In summary, our framework contains four height fields:

- Terrain H, the underneath static ground;
- Sand h^s, the granular materials;
- Pure water h^w, the part of water above the sand;
- Mixed water $h^{\bar{w}}$, the part of water submerged under sand.

We make the following assumptions to achieve real-time performance. First, sand and mixed water can reside immediately on top of the terrain. Second, pure water can reside on the terrain or mixed water but not on sand. Note that both sand and water in our framework have a constant density over the domain, i.e., ρ^s for sand and ρ^w for water. Third, different phases, including sand, pure water, and mixed water, can carry different velocities. Throughout the paper, we use superscripts s, w, and $\bar{w}$ to denote the sand, pure water, and mixed water phases, respectively. Finally, we also assume the mixed water is exactly the part of water submerged under the sand, i.e., mixed water height is an induced variable defined

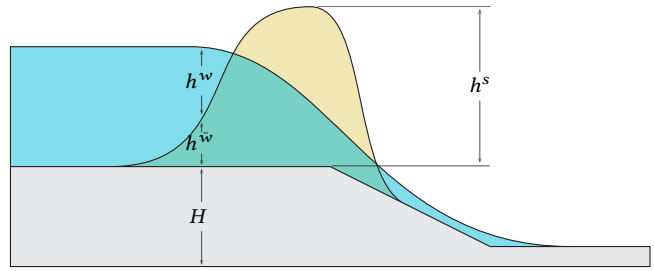


Figure 2: *The four phases in the geometric setup of our height-field simulation of sand-water mixture. Pure water h^w resides on top of and applies pressure forces on mixed water $h^{\bar{w}}$ and sand phase h^s.*

by sand and water height, $h^{\bar{w}} = \min(h^{\bar{w}} + h^w, h^s)$, where $h^{\bar{w}} + h^w$ represents the height of the entire water column.

3.2 Two-layer SWEs

Due to the shallow water assumption [Randall 2006], the conservation of mass and momentum leads to the following single-layer SWE governing equation [Bridson 2015]:

$$\frac{Dh}{Dt} = -h\nabla \cdot \mathbf{v}, \quad \frac{D\mathbf{v}}{Dt} = -g\nabla\eta + \mathbf{a},$$

where h is the water depth, H is the terrain height, D/Dt represents the material derivative, $\eta = H + h$ is the water level, g is gravity, and $\mathbf{a}$ is an external acceleration. However, our water is divided into pure and mixed phases, so we apply the conservation of mass for the two phases separately, leading to:

$$\frac{Dh^w}{Dt} = -h^w\nabla \cdot \mathbf{v}^w, \tag{1}$$

$$\frac{Dh^{\bar{w}}}{Dt} = -h^{\bar{w}}\nabla \cdot \mathbf{v}^{\bar{w}} + c_d\nabla \cdot \nabla h^{\bar{w}}, \tag{2}$$

where $\nabla \cdot \nabla$ is the Laplace operator for modeling the diffusion of water in porous media [Iaffaldano et al. 2006] and c_d is the diffusion

Table 1: Notation. $\mathcal{X}$ could be sand, mixed water or pure water.

Quantity	Description
$h^{\mathcal{X}}$	Layer height of $\mathcal{X}$
$\mathbf{v}^{\mathcal{X}}$	Layer velocity of $\mathcal{X}$
$\mathbf{F}$	Deformation gradient
$\theta/\mathbf{R}$	Rotation angle/matrix
$\mathbf{P}$	Stretch matrix
H	Terrain height
η	Water level
$\mathcal{U}$	Upwinding flux
c_e	Momentum exchange rate
c_d	Diffusion rate
$\rho^{\mathcal{X}}$	Mass density of $\mathcal{X}$
$c_0, c_1, c_2, \phi_1, \phi_2, \phi_3$	Piecewise linear function parameters
$\mathbf{a}^{\mathcal{X}}_{\mathcal{M}}$	Momentum-exchange-driven acceleration on $\mathcal{X}$
$\mathbf{a}^{\mathcal{X}}_{\mathcal{D}}$	Diffusion-driven acceleration exerted upon $\mathcal{X}$
$\mathbf{a}^{\mathcal{X}}_{\mathcal{I}}$	Internal-force-driven acceleration exerted upon $\mathcal{X}$
$\mathbf{a}^{\mathcal{X}}_{\mathcal{E}}$	Friction-driven acceleration exerted upon $\mathcal{X}$
$\mathcal{M}^{a\to b}$	Momentum exchange from a to b

rate. Regarding the conservation of momentum, we follow the two-layer SWEs [Salmon 2002], given by:

$$\frac{D\mathbf{v}^w}{Dt} = -g\nabla(H + h^w + h^{\bar{w}}) + \mathbf{a}^w_{\mathcal{M}}, \tag{3}$$

$$\frac{D\mathbf{v}^{\bar{w}}}{Dt} = -g\nabla(H + h^w + h^{\bar{w}}) + \mathbf{a}^{\bar{w}}_{\mathcal{M}} + \mathbf{a}^{\bar{w}}_{\mathcal{D}}, \tag{4}$$

where $\mathbf{a}^\star_{\mathcal{M}}$ indicates the acceleration caused by momentum exchange *and the superscript* $\star$ *can be* w *or* $\bar{w}$. $\mathbf{a}^{\bar{w}}_{\mathcal{D}}$ indicates the acceleration caused by mixed water diffusion, which only happens within the granular media between mixed water and sand.

3.3 Height-field Sand Simulation

We borrow the recent progress in SSE [Zhu et al. 2021], where the frictional term $\mathbf{a}^s_{\mathcal{E}}$ is added to the equation of momentum conservation to model the external sand-ground friction. The sand phase is also subject to two additional terms: the internal elastoplastic force $\mathbf{a}^s_{\mathcal{I}}$ between sand grains and the momentum $\mathbf{a}^s_{\mathcal{M}}$ exchanged from the mixed water phase. Put together, the conservation of mass and momentum in SSE yields:

$$\frac{Dh^s}{Dt} = -h^s\nabla\cdot\mathbf{v}^s, \tag{5}$$

$$\frac{D\mathbf{v}^s}{Dt} = -g\nabla(H + h^s + \frac{\rho^w}{\rho^s}h^w) + \mathbf{a}^s_{\mathcal{E}} + \mathbf{a}^s_{\mathcal{I}} + \mathbf{a}^s_{\mathcal{M}}, \tag{6}$$

where the term $\rho^w/\rho^s h^w$ expresses the pressure of pure water on the sand phase, accounting for the difference in their densities. We detail the various external force terms below.

3.3.1 External Frictional Force. The frictional force $\mathbf{f}_{\mathcal{E}}$ damps the relative motion between sand and the terrain underneath, which is opposite to the movement direction and proportional to the normal contact force $\mathbf{f}_C$ between the two. Applying the Coulomb's law, we have $\mathbf{f}_{\mathcal{E}} = -\mu g\mathbf{f}_C\hat{\mathbf{v}}$ and $\mathbf{f}_C = \rho^s h^s - \rho^w h^{\bar{w}}$, where $\hat{\mathbf{v}} = \mathbf{v}^s/\|\mathbf{v}^s\|$ is velocity direction, μ is the frictional coefficient, and the second term in $\mathbf{f}_C$ arises from buoyancy. Indeed, the lifting force from water decreases the normal force, thus the friction.

3.3.2 Elastoplastic Internal Force. To model the internal friction between sand grains, we use the Drucker-Prager model [Klár et al. 2016] to represent the relation between the shear and normal stresses of the sand as a continuum. To capture the cohesive effects of the saturation, we adopt modified Drucker-Prager yielding condition [Tampubolon et al. 2017]:

$$\boldsymbol{\sigma} = \frac{1}{\det(\mathbf{F})}\frac{\partial\psi}{\partial\mathbf{F}}\mathbf{F}^T, \quad c_{\mathrm{f}}\mathrm{tr}(\boldsymbol{\sigma}) + \left\|\boldsymbol{\sigma} - \frac{\mathrm{tr}(\boldsymbol{\sigma})}{d}\mathbf{I}\right\| \le c_{\mathrm{c}}(\phi),$$

where $\boldsymbol{\sigma}$ is the Cauchy stress, $\mathbf{F}$ is the deformation gradient, and ψ is the elastic energy density *adopted from Klár et al. [2016]*. c_{f} and c_{c} model the friction and level of cohesion between grains, respectively. c_{c} is a function of water saturation level ϕ in the sand, approximated by the volume fraction of water in the mixture: $\phi = (h^w + h^{\bar{w}})/(h^w + h^{\bar{w}} + h^s)$. Following the shallow assumption, the vertical velocity is negligible and we only keep track of the horizontal deformation gradient field $\mathbf{F} \in \mathbb{R}^{2\times 2}$, which evolves according to: $D\mathbf{F}/Dt = \nabla_{\mathbf{x}}\mathbf{v}\mathbf{F}$. Finally, the depth-integrated elastoplastic internal force $\mathbf{f}_{\mathcal{I}}$ reads:

$$\mathbf{f}_{\mathcal{I}} = \left(\frac{\partial(h^s\boldsymbol{\sigma}_{xx})}{\partial x} + \frac{\partial(h^s\boldsymbol{\sigma}_{xy})}{\partial y}, \frac{\partial(h^s\boldsymbol{\sigma}_{yx})}{\partial x} + \frac{\partial(h^s\boldsymbol{\sigma}_{yy})}{\partial y}\right).$$

The momentum contribution from the external friction force and elastoplastic internal force will be averaged on the whole column of sand as $\mathbf{a}_{\mathcal{E}} = \mathbf{f}_{\mathcal{E}}/(\rho^s h^s)$ and $\mathbf{a}_{\mathcal{I}} = \mathbf{f}_{\mathcal{I}}/(\rho^s h^s)$.

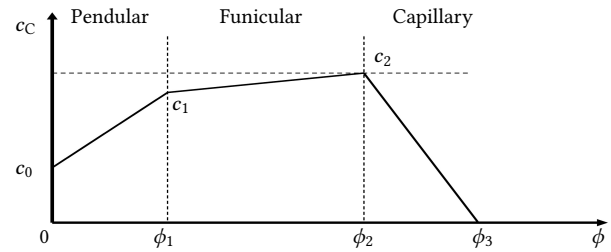


Figure 3: *A piecewise linear function is adopted to model $c_{\mathrm{c}}(\phi)$ regarding the change of saturation states of wet sand.*

3.3.3 Saturation States of Wet Sand. Dry sand has specific repose angles; however, adding a small amount of water makes the sand more cohesive, while adding a large amount of water makes the sand collapse. Microscopically, with the increase in water content, the morphology of the water phase in the sand can be categorized as *pendular*, *funicular*, and *capillary* [Wang et al. 2018]. In the pendular state (usually the level of saturation is less than 10%), isolated water bridges are formed between grains and produce attractive forces that raise the cohesion significantly. With the increase in water content, the water bridges begin to coalesce with each other to form liquid clusters and reduce the capillary force slightly. This is canceled out by the extension of rupture distance and leads to constant capillary cohesion. When the sand and water reach the capillary state of full saturation, the cohesion is reduced to about zero. We model this phenomenon using a piecewise linear function as illustrated in Fig. 3:

$$c_{\mathrm{c}}(\phi) = \begin{cases} c_0 + \phi\frac{c_1 - c_0}{\phi_1}, & \text{if } \ 0 \le \phi < \phi_1 \\ c_1 + (\phi - \phi_1)\frac{c_2 - c_1}{\phi_2 - \phi_1}, & \text{if } \phi_1 \le \phi < \phi_2 \\ c_2\frac{\phi_3 - \phi}{\phi_3 - \phi_2}, & \text{if } \phi_2 \le \phi \le \phi_3 \\ 0, & \text{if } \phi > \phi_3. \end{cases}$$

3.4 Sand and Fluid Coupling

We consider the coupling between sand and fluid in two ways: diffusion and momentum exchange.

3.4.1 Diffusion. Water can diffuse to the neighboring space within wet sand, leading to additional mass and velocity exchange. As mentioned in Sec. 3.2, mass diffusion is modeled by the term $c_d\nabla\cdot\nabla h^{\bar{w}}$ in Eq. 1,2. Similarly, the acceleration caused by the depth-integrated velocity diffusion within the porous sand can be described as: $\mathbf{a}^{\bar{w}}_{\mathcal{D}} = c_d\left(\nabla\cdot\nabla(h^{\bar{w}}\mathbf{v}^{\bar{w}})\right)/h^{\bar{w}}$, where c_d is the user-defined diffusion rate.

3.4.2 Momentum Exchange. Momentum exchange happens when water and sand flow through each other and is modeled as the additional acceleration term in the momentum conservation Eq. 6. Considering a mixed water column $h^{\bar{w}}$ collocated with a sand column h^s, the amount of momentum exchange is:

$$\mathcal{M}^{s\to\bar{w}} = -\mathcal{M}^{\bar{w}\to s} = c_e h^{\bar{w}}(\mathbf{v}^s - \mathbf{v}^{\bar{w}}),$$

where c_e is the exchange rate and $\mathcal{M}^{\bar{w}\to s}$ is the amount of momentum transferred from mixed water $\bar{w}$ to sand s phase per unit time and horizontal area.

A second type of momentum exchange can happen due to our two-layer shallow water assumptions. Since $h^{\bar{w}}$ is an induced variable of total water height $h^{\bar{w}} + h^{w}$ sand height h^{s}, some parts of mixed water can become pure water and vice versa. As the two water phases carry different velocities, there must be momentum exchange between them, which is formulated as:

$$\mathcal{M}^{w\to\bar{w}} = -\mathcal{M}^{\bar{w}\to w} = \rho^{w}\Big(\max(\tfrac{\partial h^{s}}{\partial t},0)\mathbf{v}^{w} + \min(\tfrac{\partial h^{s}}{\partial t},0)\mathbf{v}^{\bar{w}}\Big),$$

where the formula in the bracket is essentially an up-winding scheme for selecting velocity according to height field flow direction. The acceleration caused by the momentum exchange will be averaged over the whole column of the sand, pure water, and mixed water, respectively, giving: $\mathbf{a}^{s}_{\mathcal{M}} = \mathcal{M}^{\bar{w}\to s}/(\rho^{s}h^{s})$, $\mathbf{a}^{w}_{\mathcal{M}} = \mathcal{M}^{\bar{w}\to w}/(\rho^{w}h^{w})$, and $\mathbf{a}^{\bar{w}}_{\mathcal{M}} = (\mathcal{M}^{s\to\bar{w}} + \mathcal{M}^{w\to\bar{w}})/(\rho^{w}h^{\bar{w}})$.

4 NUMERICAL ALGORITHM

In this section, we describe our spatial and temporal discretization scheme for the governing equations. For spatial discretization, we use a Marker-and-Cell (MAC) [Harlow and Welch 1965] grid to store each height field, i.e., height and deformation gradient are stored at cell centers, and velocities $\mathbf{v}^{\star} \triangleq (u^{\star}, v^{\star})$ are stored at face centers. We use the subscript i, j to denote the i, jth cell center, and the fractional index indicates the face center, i.e., $i + 1/2, j$ indicates the east face center of the i, jth cell. We also use Δx and Δt to denote the cell size and the time step size, respectively. For temporal discretization, we adopt the splitting scheme. It has been pointed out in [Chentanez and Müller 2010] that the conservation of mass is visually more crucial than the conservation of momentum. To improve the stability of our scheme while maintaining visual quality, we use conservative discretization for mass exchange. Such conservative update can be conveniently implemented on a typical MAC grid using the up-winding advection scheme [Chentanez and Müller 2010].

4.1 Fluid Time Integration

Our fluid time integration algorithm is based on the splitting scheme consisting of the following steps.

4.1.1 Diffusion. We use the finite difference scheme to discretize the Laplacian operator and explicit Euler for time integration, resulting in the following discrete-time mass diffusion equation:

$$h^{\bar{w}}_{i,j} \leftarrow h^{\bar{w}}_{i,j} + c_d\tfrac{\Delta t}{\Delta x^2}(h^{\bar{w}}_{i+1,j} + h^{\bar{w}}_{i-1,j} + h^{\bar{w}}_{i,j+1} + h^{\bar{w}}_{i,j-1} - 4h^{\bar{w}}_{i,j}).$$

For the cell neighboring the domain boundary, we use the Neumann boundary condition $\nabla_{\mathbf{n}}h^{\bar{w}} = 0$, where subscript $\mathbf{n}$ denotes the normal direction at the boundary. We further set out-of-bound $h^{\bar{w}}$ to be equal to the value of its nearest neighbor. The above scheme is clearly a conservative discretization of $Dh^{\bar{w}}/Dt = c_d\nabla\cdot\nabla h^{\bar{w}}$. The induced momentum exchange is discretized in the same manner, leading to the following update rule for $u^{\bar{w}}$:

$$\begin{aligned}u^{\bar{w}}_{i+1/2,j} \leftarrow u^{\bar{w}}_{i+1/2,j} + c_d\frac{\Delta t}{\Delta x^2}\Big(&u^{\bar{w}}_{i+1,j}(h^{\bar{w}}_{i+3/2,j} - h^{\bar{w}}_{i+1/2,j}) - u^{\bar{w}}_{i,j}(h^{\bar{w}}_{i+1/2,j} - h^{\bar{w}}_{i-1/2,j})\\ &+ u^{\bar{w}}_{i+1/2,j+1/2}(h^{\bar{w}}_{i+1/2,j+1} - h^{\bar{w}}_{i+1/2,j}) - u^{\bar{w}}_{i+1/2,j-1/2}(h^{\bar{w}}_{i+1/2,j} - h^{\bar{w}}_{i+1/2,j-1})\Big),\end{aligned}$$

and a symmetric equation applies to $v^{\bar{w}}$. Since momentum does not need to be exactly conserved, we approximate face-centered height and cell-centered velocity with bilinear interpolation, so the above equation is a non-conservative discretization of $D\mathbf{v}^{\bar{w}}/Dt = \mathbf{a}^{\bar{w}}_{\mathcal{D}}$.

4.1.2 Height Integration. We use a predictor-corrector scheme to discretize the mass conservation equation $Dh^{\star}/Dt = -h\nabla\cdot\mathbf{v}^{\star}$ and time integrate the water height-fields $h^{\star}$. In particular, we first use the following conservative up-winding advection scheme [Chentanez and Müller 2010] (predictor) to update h^{w} and $h^{\bar{w}}$ separately:

$$\begin{aligned}h^{\star}_{i,j} \leftarrow h^{\star}_{i,j} &+ \tfrac{\Delta t}{\Delta x}\Big(\mathcal{U}_{i+1/2,j}(h^{\star}) - \mathcal{U}_{i-1/2,j}(h^{\star})\Big)\\ &+ \tfrac{\Delta t}{\Delta x}\Big(\mathcal{U}_{i,j+1/2}(h^{\star}) - \mathcal{U}_{i,j-1/2}(h^{\star})\Big),\end{aligned} \tag{7}$$

where $\star$ can be w or $\bar{w}$ with up-winding flux $\mathcal{U}(h)$ defined as:

$$\mathcal{U}_{i+1/2,j}(h^{\star}) = \max(u^{\star}_{i+1/2,j},0)h^{\star}_{i,j} + \min(u^{\star}_{i+1/2,j},0)h^{\star}_{i+1,j},$$
$$\mathcal{U}_{i,j+1/2}(h^{\star}) = \max(v^{\star}_{i,j+1/2},0)h^{\star}_{i,j} + \min(v^{\star}_{i,j+1/2},0)h^{\star}_{i,j+1}.$$

Such predicted height-field is not final because water can flow from pure to mixed phase and vice versa, which will also induce momentum exchanges between the two phases. Such momentum exchanges take the following form:

$$\begin{aligned}\left[\mathcal{M}^{w\to\bar{w}}\right]_{i+1/2,j} \leftarrow &\tfrac{\rho^{w}}{\Delta t}\left[\max(\min(h^{\bar{w}} + h^{w}, h^{s}) - h^{\bar{w}},0)u^{w}\right]_{i+1/2,j}\\ &+\tfrac{\rho^{w}}{\Delta t}\left[\min(\min(h^{\bar{w}} + h^{w}, h^{s}) - h^{\bar{w}},0)u^{\bar{w}}\right]_{i+1/2,j},\end{aligned}$$

where a symmetric equation applies to $\left[\mathcal{M}^{w\to\bar{w}}\right]_{i,j+1/2}$. After computing the momentum exchange, our corrector updates mixed water height-field via the induced equation: $h^{\bar{w}} \leftarrow \min(h^{\bar{w}} + h^{w}, h^{s})$.

4.1.3 Velocity Integration. Once we compute the momentum exchange between two water phases, we can update the velocity as $D\mathbf{v}^{\star}/Dt = -g\nabla(H + h^{w} + h^{\bar{w}}) + \mathbf{a}^{\star}_{\mathcal{M}}$. For time discretization, we first use the non-conservative, but unconditionally stable Semi-Lagrangian advection scheme to update the velocities. We then explicitly update velocities by taking the gradient of the water height field into account as:

$$u^{\star}_{i+1/2,j} \leftarrow u^{\star}_{i+1/2,j} + \Delta t[a^{\star}_{\mathcal{M}}]_{i+1/2,j} - \Delta tg\frac{[H + h^{w} + h^{\bar{w}}]_{i+1,j} - [H + h^{p} + h^{\bar{w}}]_{i,j}}{\Delta x},$$

and the case with $v^{\star}_{i,j+1/2}$ is symmetric.

4.2 Sand Time Integration

Similar to the fluid phase, we time integrate the sand phase using a splitting scheme, consisting of the following four steps.

4.2.1 Height Integration. We begin by re-using Eq. 7 to update the sand height-field, i.e. setting $\star$ to s.

4.2.2 Deformation Gradient Evolution. To apply elastoplastic internal force, we need to evolve $\mathbf{F}$ and then evaluate stress-induced acceleration $\mathbf{a}_{\mathcal{I}}$ from $\mathbf{F}$. However, we found that the internal force is much stiffer than other form terms, potentially leading to unstable simulation. To stabilize our simulation, we adopt the rotation-strain representation proposed in [Pan et al. 2015]. Specifically, *in order to avoid stability issues caused by direct interpolating deformation gradient,* we first perform a *polar decomposition, s.t.,* $\mathbf{F} = \mathbf{RP}$, where $\mathbf{R}$ is a rotation matrix, while $\mathbf{P}$ is a symmetric matrix. Since we only consider 2D deformation gradient, $\mathbf{R}$ is a 2D rotation, and we can safely rewrite it as a rotation angle θ. In summary, $\mathbf{F}$ is equivalently represented as θ and $\mathbf{P}$. Our main idea is to advect θ and $\mathbf{P}$ separately and then reconstruct $\mathbf{F}$.

Specifically, we first decompose $\mathbf{F}$ into θ and $\mathbf{P}$. We then use conservative up-winding advection scheme to discretize $D(h\theta)/Dt = 0$ and $D(h\mathbf{P})/Dt = 0$ as in Eq. 7. Notably, unlike the velocity advected via semi-Lagrangian, we use the conservative advection scheme for the deformation gradient, which is again conveniently doable due to the MAC grid storing $\mathbf{F}$ and $\mathbf{v}$ at the cell and face center, respectively. *Conservative advection incorporates the height of the sand, so taller sand columns (larger sand mass) are more resistant to external impacts.* After updating θ and $\mathbf{P}$, we can reconstruct $\mathbf{F}$ and use it to calculate the internal acceleration $\mathbf{a}^s_{\mathcal{I}}(\mathbf{F})$. Finally, we update sand velocity as well as deformation gradient via:

$$\mathbf{v}^s \leftarrow \mathbf{v}^s + \Delta t \mathbf{a}^s_{\mathcal{I}}(\mathbf{F}), \quad \mathbf{F} \leftarrow \mathbf{F} + \Delta t \nabla_{\mathbf{x}} \mathbf{v}^s \mathbf{F}.$$

4.2.3 Velocity Integration. We next integrate the sand velocity according to $D\mathbf{v}^s/Dt = -g\nabla(H + h^s + \rho^w h^w/\rho^s) + \mathbf{a}^s_{\mathcal{M}}$, which is time discretized in the same way as for the water, i.e., we first use Semi-Lagrangian advection to account for material derivative and then explicitly update sand velocity according to:

$$u^s_{i+1/2,j} \leftarrow u^s_{i+1/2,j} + \Delta t [a^s_{\mathcal{M}}]_{i+1/2,j} - \Delta t g \frac{[H+h^s+\frac{\rho^w}{\rho^s}h^w]_{i+1,j} - [H+h^s+\frac{\rho^w}{\rho^s}h^w]_{i,j}}{\Delta x},$$

and symmetric equations apply to the v^s component. In the above equation, the acceleration due to momentum exchange between the mixed water and the sand can be calculated via:

$$\left[a^s_{\mathcal{M}}\right]_{i+1/2,j} \leftarrow -\frac{c_e}{\rho^s h^s}\left[h^{\bar{w}}(u^s - u^{\bar{w}})\right]_{i+1/2,j}.$$

This, in return, corrects the mixed water's velocities as:

$$u^{\bar{w}}_{i+1/2,j} \leftarrow u^{\bar{w}}_{i+1/2,j} - \Delta t \frac{\rho^s h^s}{\rho^w h^{\bar{w}}}[a^s_{\mathcal{M}}]_{i+1/2,j}.$$

The case with $[a^s_{\mathcal{M}}]_{i,j+1/2}$ and $v^{\bar{w}}_{i,j+1/2}$ is, again, symmetric.

4.2.4 Applying External Frictional Force. The frictional force can be treated as an external force according to the maximal dissipation principle. We first interpolate both h^s and $h^{\bar{w}}$ at each face center and then clamp the velocity via:

$$[a^s_{\mathcal{E}}]_{\mathrm{i+1/2,j}} = \mu g\left[\frac{\rho^s h^s - \rho^w h^{\bar{w}}}{\rho^s h^s}\right]_{i+1/2,j}$$

$$u^s_{i+1/2,j} \leftarrow u^s_{i+1/2,j}\max\left(\frac{\|\mathbf{v}^s_{i+1/2,j}\| - \Delta t[a^s_{\mathcal{E}}]_{\mathrm{i+1/2,j}}}{\|\mathbf{v}^s_{i+1/2,j}\|}, 0\right).$$

We then assign $u^s_{i+1/2,j}$ to the face center and the case with $v^s_{i,j+1/2}$ is symmetric.

4.3 Simulation Scheme of Fluid and Sand Mixtures

Due to the stiffness of the sand's internal force, a much smaller time step size is required to advance the sand. Thus, we can adapt the amount of computational effort applied to each simulated species. More specifically, we simulate the water species with a large time step, assuming sand is non-existent. We then simulate sand species T steps with $1/T$ of the water time step. The outline of our sand-water mixture simulation is summarized in Algorithm 1.

Implementation Details. During initialization, we set the rotation angle θ and the stretch matrix $\mathbf{P}$ of all cells occupied by sand as zero and identity matrix, respectively. While shallow water equations can be solved to obtain velocities inside the occupied area, velocity and deformation gradient on the untracked side of the surface are needed for accurately advecting quantities of interest using the semi-Lagrangian method and calculating the internal elastoplastic force. To this end, we extrapolate data of interest by locating the untracked cell and face centers that neighbor the tracked ones. We then compute the average tracked values within the 3×3 neighborhood and assign them to the untracked ones. *If a cell is neither tracked nor adjacent, we can safely reset its physical quantities, i.e., setting the rotation angle to zero and the stretch matrix to the identity matrix. The outline of our sand-water mixture simulation is summarized in the supplemental material.*

Algorithm 1 Sand-water mixture simulator

```
Diffuse_Mixed_Water                          ▷ 4.1.1
Integrate_Water_Height                       ▷ 4.1.2
Integrate_Water_Velocity                     ▷ 4.1.3
for i ← 1 to T do
    Integrate_Sand_Height                    ▷ 4.2.1
    Integrate_Sand_Velocity                  ▷ 4.2.3
    Project_Deformation_Gradient             ▷ 3.3.2
    Evolve_Deformation_Gradient              ▷ 4.2.2
    Compute_Momentum_Exchange                ▷ 4.1.2
    Apply_Frictional_Force                   ▷ 4.2.4
end for
```

5 RESULTS

We evaluate our sand and water mixture model via a CUDA implementation, where the results are rendered in real-time with OpenGL. All timings are measured on a 3.0 GHz Intel Core i9-13900K CPU, NVIDIA GeForce RTX 4080 GPU with 16 GB Memory. In this section, we first use several examples to test the necessity of each component in our method, followed by a few complex examples. We closely follow the results of the levee wall from the previous multi-species work [Tampubolon et al. 2017] to determine the values of ϕ_1, ϕ_2, and ϕ_3 for all the simulations in our paper. The levee wall collapses when the saturated water is about half its height, which indicates $\phi_3 = 0.33$. We choose $\phi_1 = 0.2$ and $\phi_2 = 0.25$.

Saturation. We first demonstrate sand with different saturation levels in Fig. 4 and show that sand can behave quite differently based on saturation. With a low saturation, the dry sand is fully supported by external friction and forms the angle of repose. With the increase of saturation, the force from the water bridge between the grains dominates, and the sand can preserve its shape under gravity. When sand and water reach the capillary state of full saturation, the sand collapses since the cohesion is reduced to about zero.

Elastoplastic Internal Force. The "Castle" demonstrates the necessity of our elastoplastic internal force in Fig. 5. In particular, we would like to emphasize that there is only friction force in the previous shallow sand work [Zhu et al. 2021]. Unfortunately, the low external friction force cannot hold the castle's shape, while the high friction force immobilizes and prevents the castle from deforming. With the presence of elastoplastic force, the castle can be pushed forward by the water while largely holding its shape.

Single- vs. Two-layer SWEs. The division of water into two layers is imperative due to the contrasting behaviors exhibited by these two water types. As demonstrated in Fig. 6, the single-layer model assigns the same velocity to the water above and within the sand. However,

the water above the sand, which does not make contact with the sand and is expected to flow freely, experiences sluggish movement due to the interaction between mixed water and sand. In contrast, our two-layer SWEs facilitate the rapid flow of pure water, situated above the sand and unaffected by its presence.

Flush & Canyon. Fig. 1 shows a bunch of sand letters and icons washed away by the water. Fig. 7 demonstrates an example of three sand dams in the canyon flushed by the flow, in which sand, water, and terrain are all represented as the height map. Thanks to our real-time frame rate, the supplemental video shows a real-time demo where the user can dynamically add dunes and water.

Seepage. Fig. 8 shows the water from the right inlet slowly erodes the dam. Our comprehensive framework enables the generation of diverse phenomena by finely tuning the momentum exchange rate and friction. When employing a low exchange rate and friction, water can swiftly penetrate the sand, resulting in its collapse alongside the flowing water. As we raise the exchange rate and friction, the sand becomes more resistant to collapse, yet water still seeps through its structure. By employing a high exchange rate and friction, the sand effectively acts as a barrier, impeding the passage of water. As we progressively raise the water level, the dam eventually breaks due to internal seepage erosion, and the landslide creates interesting textures in the debris flow.

Performance. Our method can be easily parallelized on a GPU and achieve a real-time frame rate. Table 2 gives the percentage of the simulation times used for different operations as well as the total computation time per step for scenes tested in the paper. T indicates how many sand updates are followed by one water update step. Over 70% of the simulation time is spent on two operations, deformation gradient projection, and boundary conditions update. The deformation gradient update takes multiple CUDA kernel passes to first extrapolate θ and $\mathbf{P}$ and velocity at the boundary faces and centers, then perform matrix SVD decomposition, and finally update and project deformation gradient matrices. Boundary condition update happens before almost every operation to determine the boundary type so the following kernels can process them accordingly. To avoid the performance impact caused by additional CUDA-parallel-reduction and CPU readback required by enforcing the CFL condition, we follow the rules of thumb in the production to not enforce CFL. Instead, we employ velocity clamping to ensure a constant time step size, thus maintaining optimal performance without compromising computational efficiency.

Rendering. Specifically, we employ three $N \times M$ height meshes for water, sand, and terrain, respectively, where N and M refer to the length and width of the 2.5D simulation domain. In each frame, we transfer the computed water and sand height data from the simulation to 2D OpenGL texture buffers and smooth the data with 3×3 Gaussian filter. These buffers are utilized as heightmaps to perturb vertices on water and sand meshes during rendering. We further incorporate a screen space ambient occlusion (SSAO) pass and a shadow pass before rendering the water and sand from a top view to a framebuffer. For sand shading, we use a real-time shading model [Edwards 2013] that considers diffuse, rim light, ocean, specular, and glitter reflection. Moreover, to differentiate between sand with different saturations, we use the saturation value to interpolate between dry and wet sand colors. The water color is also interpolated between shallow and deep water colors based on the water depth. To compute water reflection and refraction, we deploy a ray-marching technique within the 2D water/sand height texture. When the ray hits the ground, we fetch the color from the previous top-down framebuffer. We also transfer the water velocity field from the simulation to the shader as a flow map to distort the normal map in the flow direction [Vlachos 2010].

Table 2: *Performance breakdown*

		Castle	Spring	Dam Break	Seepage	Flush	Canyon
Resolution	x	512	1024	512	512	1024	600
	y	1024	1024	512	1024	1024	800
Time step size		0.009	0.012	0.045	0.015	0.027	0.03
Async coefficient T		3	3	3	3	3	3
Diffuse & integrate water		1.2%	1.7%	1.4%	0.9%	1.3%	0.8%
Integrate water velocity		1.1%	1.1%	2.0%	1.0%	0.9%	0.9%
Integrate sand height		3.8%	4.0%	3.0%	2.8%	4.0%	3.0%
Integrate sand velocity		6.2%	4.8%	6.7%	6.2%	5.2%	4.8%
Project deformation grad.		40.5%	44.5%	44.1%	40.5%	45.5%	38.4%
Evolve deformation grad.		5.7%	5.4%	8.9%	6.0%	5.7%	5.2%
Compute momentum ex.		6.1%	8.4%	6.7%	7.4%	6.7%	6.2%
Apply frictional force		2.1%	1.8%	2.8%	2.1%	1.9%	1.8%
Update boundary cond.		33.5%	28.2%	24.6%	33.1%	28.6%	38.9%
Avg. time per step (ms)		0.73	2.16	0.46	3.54	2.04	0.61

6 CONCLUSION

We have presented a real-time simulation framework for the water and sand mixture. Our framework is based on height fields representing different phases, including sand, water, and mixed water. Our method achieves a great tradeoff between fidelity and performance, which can be an excellent technique for interactive applications. Our framework formulates the external frictional force and elastoplastic internal force of sand based on the grid and also handles the water/sand coupling via diffusion and momentum exchange. The whole system can be time integrated efficiently using a semi-implicit operator splitting discretization scheme. We also demonstrate that our proposed system can be implemented efficiently on the GPU to simulate various water-sand mixture scenarios.

Our approach shares the limitations of the Shallow Water Equations, including two key factors: 1) the vertical length scale must be significantly smaller than the horizontal length scale, and 2) the accurate handling of vertical variations in velocity and force is challenging. Consequently, our method is unable to capture phenomena such as vertical layered flow, particle-laden flow, and sharp features accurately. Apparently, with the height field representation, we do not consider the sand deformation gradient changes along the vertical direction. It would be helpful to consider that vertical change in the future to simulate the castle with very sharp walls. *Its potential to generate novel visual effects for sand animation within interactive frameworks, however, is curtailed by the inherent visual expressiveness of the height-field method. Addressing this limitation presents a compelling avenue for future research.* Another interesting future work would consider the change of the sand constitutive model with the grain size.

ACKNOWLEDGMENTS

We would like to thank our colleagues from LightSpeed Studios, Fengquan Wang and Dong Li, for their project support.

REFERENCES

Iván Alduán and Miguel A. Otaduy. 2011. SPH Granular Flow with Friction and Cohesion. In *Proceedings of the 2011 ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (Vancouver, British Columbia, Canada) *(SCA '11)*. Association for Computing Machinery, New York, NY, USA, 25–32.

Nathan Bell, Yizhou Yu, and Peter J. Mucha. 2005. Particle-Based Simulation of Granular Materials. In *Proceedings of the 2005 ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (Los Angeles, California) *(SCA '05)*. Association for Computing Machinery, New York, NY, USA, 77–86.

B. Benes and R. Forsbach. 2001. Layered data representation for visual simulation of terrain erosion. In *Proceedings Spring Conference on Computer Graphics*. IEEE, usa, 80–86.

Bedrich Benes, Václav Těšínský, Jan Hornyš, and Sanjiv K. Bhatia. 2006. Hydraulic erosion. *Computer Animation and Virtual Worlds* 17, 2 (2006), 99–108.

Robert Bridson. 2015. *Fluid simulation for computer graphics*. AK Peters/CRC Press, New York, NY, USA.

Nuttapong Chentanez and Matthias Müller. 2010. Real-Time Simulation of Large Bodies of Water with Small Scale Details. In *Proceedings of the 2010 ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (Madrid, Spain) *(SCA '10)*. Eurographics Association, Goslar, DEU, 197–206.

Nuttapong Chentanez, Matthias Müller, and Tae-Yong Kim. 2015. Coupling 3D Eulerian, Heightfield and Particle Methods for Interactive Simulation of Large Scale Liquid Phenomena. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (Copenhagen, Denmark) *(SCA '14)*. Eurographics Association, Goslar, DEU, 1–10.

Gilles Daviet and Florence Bertails-Descoubes. 2016. A Semi-Implicit Material Point Method for the Continuum Simulation of Granular Materials. *ACM Trans. Graph.* 35, 4, Article 102 (jul 2016), 13 pages.

John Edwards. 2013. Sand Rendering in Journey. (2013). Game Developers Conference (GDC).

Yun (Raymond) Fei, Christopher Batty, Eitan Grinspun, and Changxi Zheng. 2019. A Multi-Scale Model for Coupling Strands with Shear-Dependent Liquid. *ACM Trans. Graph.* 38, 6, Article 190 (nov 2019), 20 pages.

Ming Gao, Andre Pradhana, Xuchen Han, Qi Guo, Grant Kot, Eftychios Sifakis, and Chenfanfu Jiang. 2018a. Animating Fluid Sediment Mixture in Particle-Laden Flows. *ACM Trans. Graph.* 37, 4, Article 149 (jul 2018), 11 pages.

Ming Gao, Xinlei Wang, Kui Wu, Andre Pradhana, Eftychios Sifakis, Cem Yuksel, and Chenfanfu Jiang. 2018b. GPU Optimization of Material Point Methods. *ACM Trans. Graph.* 37, 6, Article 254 (dec 2018), 12 pages.

Prashant Goswami, Philipp Schlegel, Barbara Solenthaler, and Renato Pajarola. 2010. Interactive SPH Simulation and Rendering on the GPU. In *Proceedings of the 2010 ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (Madrid, Spain) *(SCA '10)*. Eurographics Association, Goslar, DEU, 55–64.

T.R. Hagen, J.M. Hjelmervik, K.-A. Lie, J.R. Natvig, and M. Ofstad Henriksen. 2005. Visual simulation of shallow-water waves. *Simulation Modelling Practice and Theory* 13, 8 (2005), 716–726. Programmable Graphics Hardware.

Francis H. Harlow and J. Eddie Welch. 1965. Numerical Calculation of Time-Dependent Viscous Incompressible Flow of Fluid with Free Surface. *The Physics of Fluids* 8, 12 (12 1965), 2182–2189.

Xiaowei He, Huamin Wang, and Enhua Wu. 2018. Projective Peridynamics for Modeling Versatile Elastoplastic Materials. *IEEE Transactions on Visualization and Computer Graphics* 24, 9 (2018), 2589–2599.

G Iaffaldano, M Caputo, and S Martino. 2006. Experimental and theoretical memory diffusion of water in sand. *Hydrology and Earth System Sciences* 10, 1 (2006), 93–100.

Stefan Jeschke and Chris Wojtan. 2017. Water wave packets. *ACM Transactions on Graphics (TOG)* 36, 4 (2017), 1–12.

Gergely Klár, Theodore Gast, Andre Pradhana, Chuyuan Fu, Craig Schroeder, Chenfanfu Jiang, and Joseph Teran. 2016. Drucker-Prager Elastoplasticity for Sand Animation. *ACM Trans. Graph.* 35, 4, Article 103 (jul 2016), 12 pages.

T. Koike, S. Morishima, and R. Ando. 2020. Asynchronous Eulerian Liquid Simulation. *Computer Graphics Forum* 39, 2 (2020), 1–8.

Anita T Layton and Michiel van de Panne. 2002. A numerically efficient and stable algorithm for animating water waves. *The Visual Computer* 18, 1 (2002), 41–53.

Toon Lenaerts and Philip Dutré. 2009. Mixing Fluids and Granular Materials. *Computer Graphics Forum* 28, 2 (2009), 213–218.

David I. W. Levin, Joshua Litven, Garrett L. Jones, Shinjiro Sueda, and Dinesh K. Pai. 2011. Eulerian Solid Simulation with Contact. *ACM Trans. Graph.* 30, 4, Article 36 (jul 2011), 10 pages.

A. Luciani, A. Habibi, and E. Manzotti. 1995. A Multi-Scale Physical Model of Granular Materials. In *Proceedings of Graphics Interface '95* (Quebec, Quebec, Canada) *(GI '95)*. Canadian Human-Computer Communications Society, Toronto, Ontario, Canada, 136–146.

Rahul Narain, Abhinav Golas, and Ming C. Lin. 2010. Free-Flowing Granular Materials with Two-Way Solid Coupling. In *ACM SIGGRAPH Asia 2010 Papers* (Seoul, South Korea) *(SIGGRAPH ASIA '10)*. Association for Computing Machinery, New York, NY, USA, Article 173, 10 pages.

Koichi Onoue and Tomoyuki Nishita. 2005. An Interactive Deformation System for Granular Material. *Computer Graphics Forum* 24, 1 (2005), 51–60.

Zherong Pan, Hujun Bao, and Jin Huang. 2015. Subspace dynamic simulation using rotation-strain coordinates. *ACM Transactions on Graphics (TOG)* 34, 6 (2015), 1–12.

David A Randall. 2006. The shallow water equations. *Department of Atmospheric Science, Colorado State University, Fort Collins* 1, 1 (2006), 52 pages.

Bo Ren, Ben Xu, and Chenfeng Li. 2021. Unified Particle System for Multiple-Fluid Flow and Porous Material. *ACM Trans. Graph.* 40, 4, Article 118 (jul 2021), 14 pages.

Witawat Rungjiratananon, Zoltan Szego, Yoshihiro Kanamori, and Tomoyuki Nishita. 2008. Real-time Animation of Sand-Water Interaction. *Computer Graphics Forum* 27, 7 (2008), 7 pages.

Rick Salmon. 2002. Numerical solution of the two-layer shallow water equations with bottom topography. *Journal of marine research* 60, 4 (2002), 605–638.

Barbara Solenthaler and Markus Gross. 2011. Two-Scale Particle Simulation. In *ACM SIGGRAPH 2011 Papers* (Vancouver, British Columbia, Canada) *(SIGGRAPH '11)*. Association for Computing Machinery, New York, NY, USA, Article 81, 8 pages.

Jos Stam. 1999. Stable Fluids. In *Proceedings of the 26th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '99)*. ACM Press/Addison-Wesley Publishing Co., USA, 121–128.

Robert W Sumner, James F O'Brien, and Jessica K Hodgins. 1999. Animating sand, mud, and snow. *Computer Graphics Forum* 18, 1 (1999), 17–26.

Tetsuya Takahashi and Christopher Batty. 2021. FrictionalMonolith: A Monolithic Optimization-Based Approach for Granular Flow with Contact-Aware Rigid-Body Coupling. *ACM Trans. Graph.* 40, 6, Article 206 (dec 2021), 20 pages.

Andre Pradhana Tampubolon, Theodore Gast, Gergely Klár, Chuyuan Fu, Joseph Teran, Chenfanfu Jiang, and Ken Museth. 2017. Multi-Species Simulation of Porous Sand and Water Mixtures. *ACM Trans. Graph.* 36, 4, Article 105 (jul 2017), 11 pages.

Yun Teng, David I. W. Levin, and Theodore Kim. 2016. Eulerian Solid-Fluid Coupling. *ACM Trans. Graph.* 35, 6, Article 200 (dec 2016), 8 pages.

Nils Thurey, Matthias Muller-Fischer, Simon Schirm, and Markus Gross. 2007. Real-time Breaking Waves for Shallow Water Simulations. In *15th Pacific Conference on Computer Graphics and Applications (PG'07)*. IEEE, Maui, HI, USA, 39–46.

Adrien Treuille, Andrew Lewis, and Zoran Popović. 2006. Model reduction for real-time fluids. *ACM Transactions on Graphics (TOG)* 25, 3 (2006), 826–834.

Alex Vlachos. 2010. Water Flow in Portal 2. (2010). Advances in Real-Time Rendering in 3D Graphics and Games in Siggraph.

Ondřej Šťava, Bedřich Beneš, Matthew Brisbin, and Jaroslav Křivánek. 2008. Interactive Terrain Modeling Using Hydraulic Erosion. In *Proceedings of the 2008 ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (Dublin, Ireland) *(SCA '08)*. Eurographics Association, Goslar, DEU, 201–210.

Ji-Peng Wang, Xia Li, and Hai-Sui Yu. 2018. A micro–macro investigation of the capillary strengthening effect in wet granular materials. *Acta Geotechnica* 13, 3 (2018), 513–533.

Xu Wang, Makoto Fujisawa, and Masahiko Mikawa. 2021. Visual Simulation of Soil-Structure Destruction with Seepage Flows. *Proc. ACM Comput. Graph. Interact. Tech.* 4, 3, Article 41 (sep 2021), 18 pages.

Xinlei Wang, Minchen Li, Yu Fang, Xinxin Zhang, Ming Gao, Min Tang, Danny M Kaufman, and Chenfanfu Jiang. 2020. Hierarchical optimization time integration for cfl-rate mpm stepping. *ACM Transactions on Graphics (TOG)* 39, 3 (2020), 1–16.

Xiao Yan, Yun-Tao Jiang, Chen-Feng Li, Ralph R. Martin, and Shi-Min Hu. 2016. Multiphase SPH Simulation for Interactive Fluids and Solids. *ACM Trans. Graph.* 35, 4, Article 79 (jul 2016), 11 pages.

Tao Yang, Jian Chang, Ming C. Lin, Ralph R. Martin, Jian J. Zhang, and Shi-Min Hu. 2017. A Unified Particle System Framework for Multi-Phase, Multi-Material Visual Simulations. *ACM Trans. Graph.* 36, 6, Article 224 (nov 2017), 13 pages.

Yonghao Yue, Breannan Smith, Peter Yichen Chen, Maytee Chantharayukhonthorn, Ken Kamrin, and Eitan Grinspun. 2018. Hybrid Grains: Adaptive Coupling of Discrete and Continuum Simulations of Granular Media. *ACM Trans. Graph.* 37, 6, Article 283 (dec 2018), 19 pages.

Cem Yuksel, Donald H House, and John Keyser. 2007. Wave particles. *ACM Transactions on Graphics (TOG)* 26, 3 (2007), 99–es.

Bo Zhu and Xubo Yang. 2010. Animating Sand as a Surface Flow. In *Eurographics 2010 - Short Papers*, H. P. A. Lensch and S. Seipel (Eds.). The Eurographics Association, Norrköping, Sweden, 4 pages.

Kuixin Zhu, Xiaowei He, Sheng Li, Hongan Wang, and Guoping Wang. 2021. Shallow Sand Equations: Real-Time Height Field Simulation of Dry Granular Flows. *IEEE Transactions on Visualization and Computer Graphics* 27, 3 (2021), 2073–2084.

Yongning Zhu and Robert Bridson. 2005. Animating Sand as a Fluid. *ACM Trans. Graph.* 24, 3 (jul 2005), 965–972.

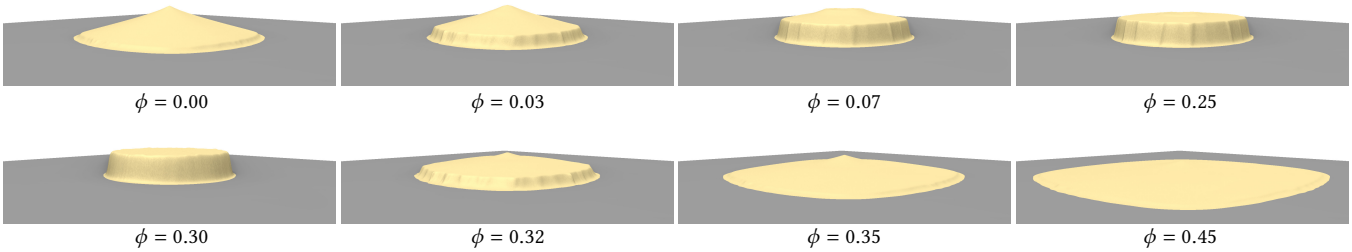


Figure 4: ***Piles:*** *Sand is initialized in a cylinder and the simulation runs until the sand reaches a stable state. The final pose is shown in the sub-figures. With different saturation levels, the cohesive force that holds the sand together increases with a higher saturation level at first and then decreases when the sand is overly saturated, while the external frictional force keeps decreasing as a result of enlarging buoyancy.*

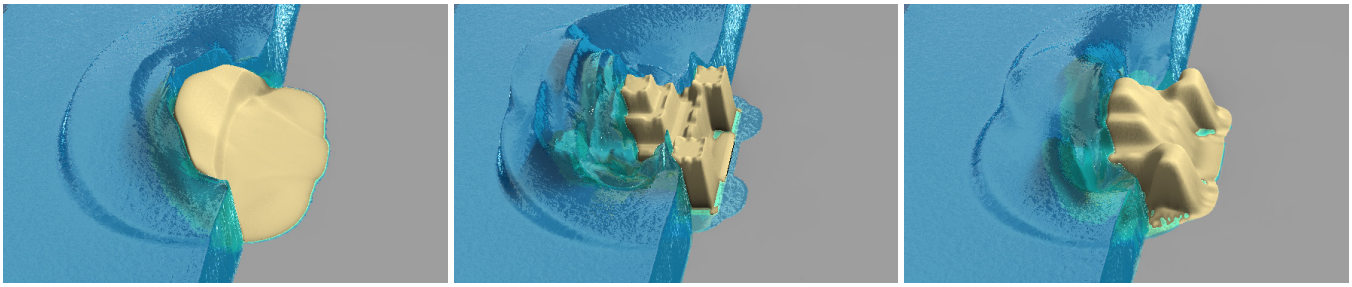


Figure 5: ***Castle:*** *To compare with the previous shallow sand method [Zhu et al. 2021], we demonstrate that using their frictional force only approach, the sand is either dispersed when the frictional coefficient is small (left) or stays still when the frictional coefficient is large (middle). Our internal elastoplastic force makes the sand be pushed forward while largely holding its shape (right).*

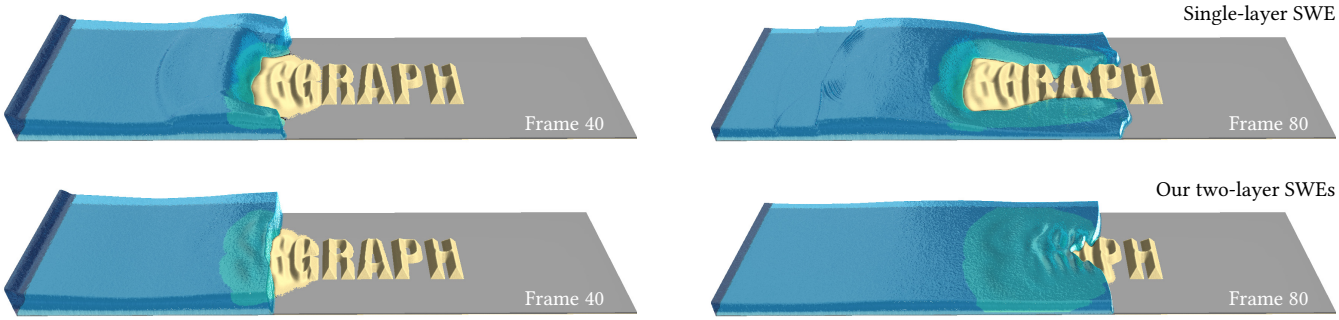


Figure 6: ***Letters:*** *The sand "SIGGRAPH" letters are washed away by the water. The top is with single-layer SWE where the movement of water on the top is impeded by the underneath sand, and the bottom is with two-layer SWEs.*

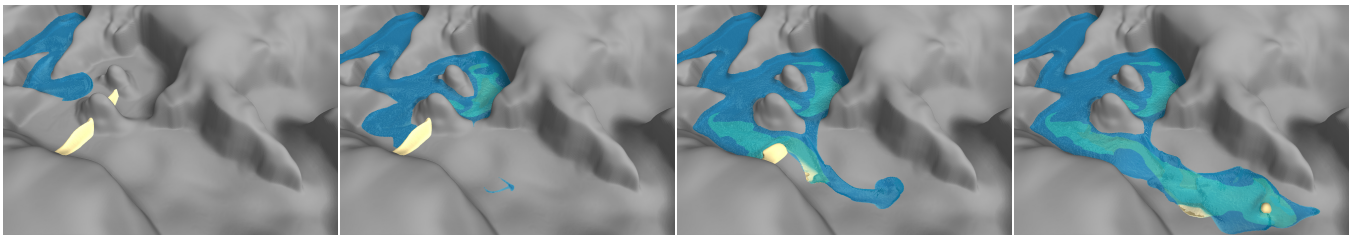


Figure 7: ***Canyon:*** *Sand dams in the canyon flushed by the flow.*

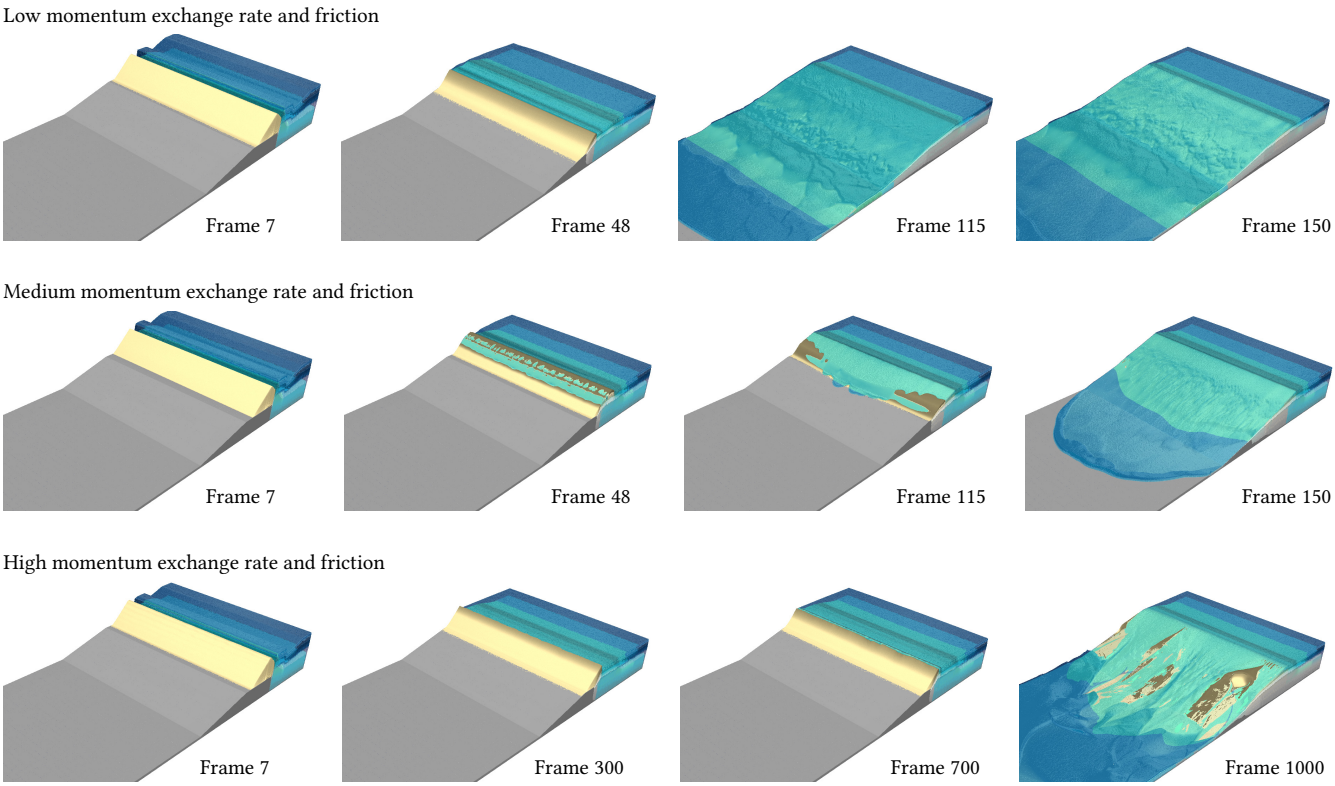


Figure 8: ***Seepage:*** *The water from the right slowly erodes the dam, which eventually breaks due to internal seepage erosion, and the landslide creates interesting textures in the debris flow. From top to bottom, by increasing momentum exchange rate and friction, various phenomena can be created, such as water leaking through the sand dam or sand behaving like a barrier before the collapse.*

Supplemental Document: Real-time Height-field Simulation of Sand and Water Mixtures

Haozhe Su
haozhesu@global.tencent.com
LightSpeed Studios
Los Angeles, CA, USA

Siyu Zhang
cuzhang@global.tencent.com
LightSpeed Studios
Los Angeles, CA, USA

Zherong Pan
zrpan@global.tencent.com
LightSpeed Studios
Seattle, WA, USA

Mridul Aanjaneya
ma635@cs.rutgers.edu
Rutgers University
New Brunswick, NJ, USA

Xifeng Gao
xifgao@global.tencent.com
LightSpeed Studios
Seattle, WA, USA

Kui Wu
kwwu@global.tencent.com
LightSpeed Studios
Los Angeles, CA, USA

ACM Reference Format:
Haozhe Su, Siyu Zhang, Zherong Pan, Mridul Aanjaneya, Xifeng Gao, and Kui Wu. 2023. Supplemental Document: Real-time Height-field Simulation of Sand and Water Mixtures. In *SIGGRAPH Asia 2023 Conference Papers (SA Conference Papers '23), December 12–15, 2023, Sydney, NSW, Australia.* ACM, New York, NY, USA, 3 pages. https://doi.org/10.1145/3610548.3618159

1 STABILITY ANALYSIS

Accurate interpolation/extrapolation of the deformation gradient $\mathbf{F}$ is crucial for stress calculation, and the naive approach of averaging nearby deformation gradients can bring stability issues. In the rest of this section, we will demonstrate the instability of direct interpolation in the simple case of pure rotation. Suppose we only consider pure rotation in 2D, in other word, the stretch matrix is a 2×2 identity matrix:

$$\mathbf{P} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \tag{1}$$

We use $\mathbf{F}_0$ and $\mathbf{F}_t$ to denote the initial and the final deformation gradient, respectively. Suppose our goal is to estimate the deformation gradient $\overline{\mathbf{F}}$ in the middle of these two states. We have two ways to interpolate the deformation gradient as:

(1) **Direct interpolation** $\overline{\mathbf{F}} = (\mathbf{F}_0 + \mathbf{F}_t)/2$
(2) **With polar decomposition** We first apply polar decomposition to both $\mathbf{F}_0$ and $\mathbf{F}_t$: $\mathbf{F}_0 = \mathbf{R}_0\mathbf{P}_0 = \mathbf{R}_0, \mathbf{F}_t = \mathbf{R}_t\mathbf{P}_t = \mathbf{R}_t$, then we can obtain the corresponding rotation angle θ_0 and θ_t. We interpolate the rotation angle by $\overline{\theta} = (\theta_0 + \theta_t)/2$ and reconstruct the full deformation gradient by

$$\overline{\mathbf{F}} = \begin{pmatrix} \cos(\overline{\theta}) & -\sin(\overline{\theta}) \\ \sin(\overline{\theta}) & \cos(\overline{\theta}) \end{pmatrix} \tag{2}$$

Next, we prove that direct interpolation may introduce extra, unnecessary compression via the following identity:

$$\begin{aligned}
\overline{\mathbf{F}} &= \frac{1}{2}(\mathbf{F}_0 + \mathbf{F}_t) \\
&= \frac{1}{2}\begin{pmatrix} \cos(\theta_0) + \cos(\theta_t) & -\sin(\theta_0) - \sin(\theta_t) \\ \sin(\theta_0) + \sin(\theta_t) & \cos(\theta_0) + \cos(\theta_t) \end{pmatrix} \\
&= \begin{pmatrix} \cos(\overline{\theta})\cos(\tilde{\theta}) & -\sin(\overline{\theta})\cos(\tilde{\theta}) \\ \sin(\overline{\theta})\cos(\tilde{\theta}) & \cos(\overline{\theta})\cos(\tilde{\theta}) \end{pmatrix} \\
&= \underbrace{\begin{pmatrix} \cos(\overline{\theta}) & -\sin(\overline{\theta}) \\ \sin(\overline{\theta}) & \cos(\overline{\theta}) \end{pmatrix}}_{\mathbf{R}} \underbrace{\begin{pmatrix} \cos(\tilde{\theta}) & 0 \\ 0 & \cos(\tilde{\theta}) \end{pmatrix}}_{\mathbf{P}}
\end{aligned} \tag{3}$$

where $\overline{\theta} = (\theta_0 + \theta_t)/2$ and $\tilde{\theta} = (\theta_0 - \theta_t)/2$. The resulting deformation gradient can be viewed as a stretch matrix $\mathbf{P} = \cos(\tilde{\theta})\mathbf{I}$, followed by a rotation matrix $\mathbf{R}$. From Equation 3, we can see that the rotation angle is correctly interpolated. However, when $\theta_0 - \theta_t \neq 2k\pi$ where k is an integer, the objective is mistakenly compressed. In the worst case ($\theta_0 - \theta_t = \pi + 2l\pi$, where l is an integer), direct interpolation results in $\overline{\mathbf{F}} = \mathbf{0}$, causing extremely compression and severe stability issues. Figure 1 compares the interpolated results of two states using direct interpolation on top as well as with polar decomposition at the bottom.

2 ASYNCHRONOUS UPDATE

To further evaluate the method of our asynchronous update, we run the same scene with different numbers of sand/water step ratios and $\Delta t = 0.004$. As the "Spring" examples shown in Figure 2 demonstrate, the asynchronous update can improve the performance by 13% and 20%, for $T = 2$ and 3, respectively, at the expense of minor changes in the results. Notably, a substantial portion of the computational time is allocated to the evolution of the sand, leading to only a modest enhancement in overall performance. In particular, we run the same scene with $\Delta t = 10^{-5}$ as the baseline and measure the Mean Absolute Error (MAE). We find the MAE of $T = 2, 3$ to be less than 2% compared with $T = 1$ on average.

3 COMPARISON WITH MPM

Eulerian-Lagrangian hybrid method, e.g., MPM [2], has been proved to successfully simulate the dynamics of sand with high fidelity at the cost of performance. To establish a comparative analysis between our method and MPM, we initially employ our approach

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

SA Conference Papers '23, December 12–15, 2023, Sydney, NSW, Australia

© 2023 Copyright held by the owner/author(s). Publication rights licensed to ACM.

ACM ISBN 979-8-4007-0315-7/23/12…$15.00
https://doi.org/10.1145/3610548.3618159

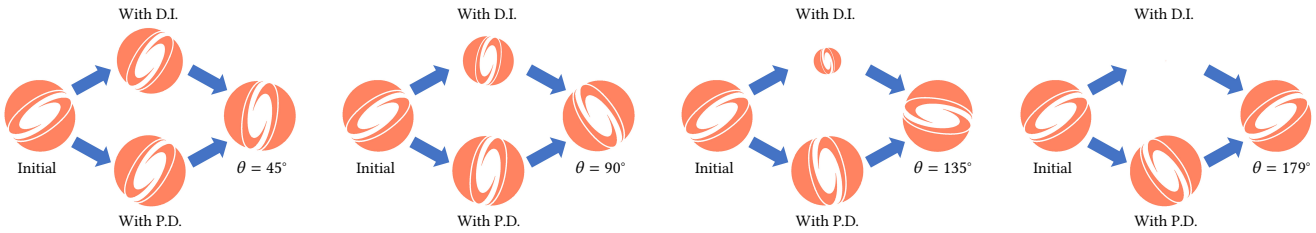
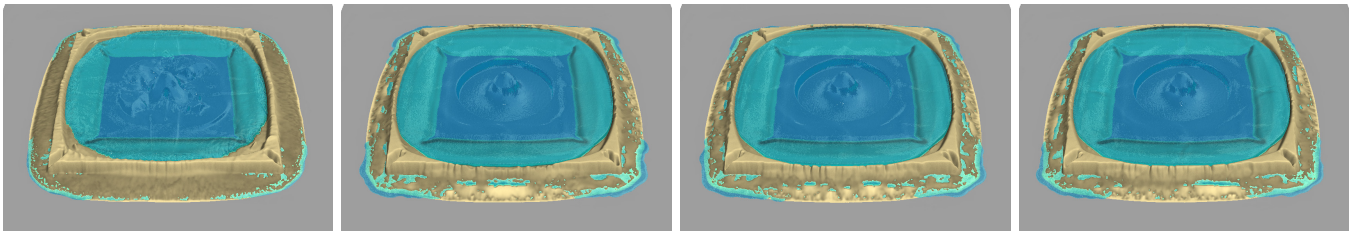


Figure 1: Comparison: In each sub-figure, the rightmost state is generated by rotating the leftmost state by θ, and the middle states on top and at the bottom are the interpolated results of the leftmost and rightmost states using direct interpolation (D.I.) of the deformation gradient F, as well as with polar decomposition (P.D.), respectively. Direct interpolation introduces extra compression and it is more and more severe as θ approaches 180°, while with polar decomposition, the accurate angle is calculated and the object size is preserved.



Baseline $\Delta t = 10^{-5}$ | $T = 1$, 2.69 ms per step | $T = 2$, 2.33 ms per step | $T = 3$, 2.16 ms per step

Figure 2: Spring: Water flows from the center and pushes surrounding sand away. The leftmost one is the baseline simulated with $\Delta t = 10^{-5}$ and the rest are with $\Delta t = 0.004$ and different sand/water update ratios T as 1, 2, and 3, respectively.

MPM 8K | MPM 500K | Ours | MPM 8K | MPM 500K | Ours

Figure 3: Comparison with 3D MPM: With the same runtime performance, our height-based sand model exhibits dynamics similar to MPM but with more details. Ours uses a 1024^2 height-based field, while MPM has around 8K particles with a 64^3 grid and 500K particles with a 256^3 grid, respectively.

to simulate a sand castle within a 1024×1024 domain, achieving a computational time of approximately $1.8ms$ per step. Subsequently, we utilize Taichi GPU MPM [1] to replicate the same scenario, with a time constraint of $1.8ms$ per step, allowing for the utilization of only 7833 particles with a 64^3 grid. Notably, both our method and MPM do not impose the CFL condition. Figure 3 demonstrates that our height-based sand model with the same runtime performance exhibits dynamics similar to MPM but with more details. In order to obtain more details, we further increase the grid resolution and particle number of MPM to 256^3 and $500K$, respectively, which needs $700ms$ per step.

4 SCALABILITY

To test the scalability, we employ a "Dam Break" example with different grid resolutions, 128^2, 256^2, 512^2, and 1024^2, respectively. Figure 4 shows our method can be easily scaled to a large scene with small memory usage. More importantly, the one with a 1024^2 grid only takes 2.11 ms per step, which makes this technique accessible to real-time applications.

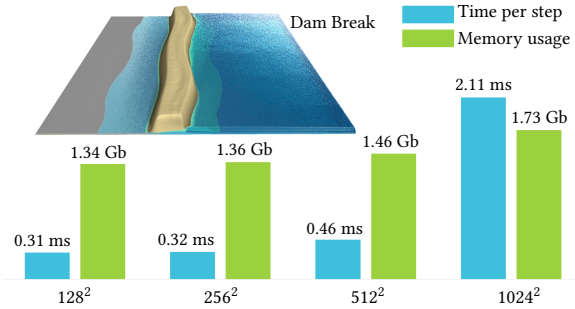


Figure 4: Performance and memory usage for the "Dam Break" example (top left) with different grid resolutions.

REFERENCES

[1] Yuanming Hu, Tzu-Mao Li, Luke Anderson, Jonathan Ragan-Kelley, and Frédo Durand. 2019. Taichi: a language for high-performance computation on spatially sparse data structures. *ACM Transactions on Graphics (TOG)* 38, 6 (2019), 201.

[2] Andre Pradhana Tampubolon, Theodore Gast, Gergely Klár, Chuyuan Fu, Joseph Teran, Chenfanfu Jiang, and Ken Museth. 2017. Multi-Species Simulation of Porous Sand and Water Mixtures. *ACM Trans. Graph.* 36, 4, Article 105 (jul 2017), 11 pages.