Eurographics Workshop on Natural Phenomena (2007)
D. Ebert, S. Mérillou (Editors)

Animating Corrosion and Erosion

Chris Wojtan,[1] Mark Carlson,[1] Peter J. Mucha,[2] and Greg Turk[1]

[1]Georgia Institute of Technology, [2]University of North Carolina, Chapel Hill

Abstract

In this paper, we present a simple method for animating natural phenomena such as erosion, sedimentation, and acidic corrosion. We discretize the appropriate physical or chemical equations using finite differences, and we use the results to modify the shape of a solid body. We remove mass from an object by treating its surface as a level set and advecting it inward, and we deposit the chemical and physical byproducts into simulated fluid. Similarly, our technique deposits sediment onto a surface by advecting the level set outward. Our idea can be used for off-line high quality animations as well as interactive applications such as games, and we demonstrate both in this paper.

Categories and Subject Descriptors (according to ACM CCS): I.3.5 [Computer Graphics]: Physically based modeling I.3.7 [Computer Graphics]: Animation I.6.8 [Computer Graphics]: Animation

1. Introduction

Fluid's ability to drastically change the form of objects is an important special effect in action movies. In "Alien," creature blood doubled as a powerful corrosive agent. In "The Wizard of Oz," a bucket of water was used to eradicate the antagonist. Artists and directors usually create such effects with trick photography or dangerous chemicals [McC91]. With the increasing dependence on computer models in such movies, these techniques can be made obsolete. In addition, seemingly tame natural phenomena become very powerful on large scales, and it is infeasible to capture these effects without using a computer simulation. The drastic topology changes to a planet over time due to erosion and sedimentation simply cannot be filmed because the process takes so long.

In this paper, we present a technique for simulating the effects of erosion, sedimentation, and corrosion at the interface between a solid and a liquid. We erode a surface by tearing away geometry and depositing it into simulated fluid. This erosion process is a physically based approximation to the complex erosion procedure studied in the geology community. Our technique deposits sediment onto a surface in a similar manner. We also address the corrosion of surfaces due to acid. We outline the necessary chemical equations and implement them using our method. In addition to our off-line simulation methods, we demonstrate an interactive application of our erosion and corrosion technique.

2. Previous Work

A significant body of work has appeared about animating physical phenomena such as fluids. We will only review the work that is most closely related to our own.

2.1. Changes at the Solid-Fluid Interface

Melek and Keyser [MK03] treated their solid objects as level sets, and they simulated burning by advecting the level sets inward. We similarly remove geometry using level sets, but we define the surface velocity differently depending on the phenomena being simulated. Our idea is also similar in spirit to that of Losasso *et al.* [LIGF06]. Their work combines the two-way solid-fluid coupling of Guendelman *et al.* [GSLF05] with shape-modification techniques like that of Melek and Keyser [MK03] in order to produce burning and melting effects. Our work uses similar ideas to simulate other reactions at the solid-fluid interface, namely erosion, sedimentation, and corrosion.

When our simulations require solid-fluid coupling, we use the rigid fluid method [CMT04], because it is fast and easy to implement. We treat our solid objects as rigid bodies, but we allow drastic geometry changes to all of the solids in the simulation. We could easily use the coupling method of [GSLF05], [CGFO06], or [BBB07] if a simulation required more accurate coupling or deformable bodies.

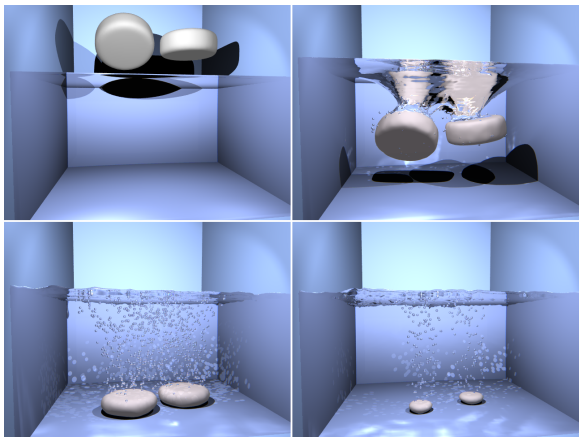

Figure 1: *Acid combines with baking soda to produce carbon dioxide gas.*

2.2. Simulation of Erosion and Chemical Processes

A significant amount of research has been done to simulate erosion on surfaces. Musgrave *et al*. [MKM89] computed long-term erosion due to thermal and hydraulic effects, and applied their work to fractal height fields to simulate erosion on terrain. Dorsey *et al*. [DEL*99] achieved stunning results by simulating erosion to produce weathered stone. More recently, Benes *et al*. [BTHB06] modeled the drastic geometry changes of hydraulic erosion and produced impressive sedimentation simulations using a material transport equation. However, their model does not support changes in the topology of the eroded surface. Merillou *et al*. [MDG01] and Chang *et al*. [CS03] produced detailed work on corrosion. These methods are accurate and beautiful, but they cannot handle drastic geometry changes.

Recent work in chemical kinetics [IKC04] gives an in-depth description of how to simulate chemical reactions between gases to produce eye-catching animations. Cutler *et al*. [CDM*02] use simulation as a modeling tool to create a vast collection of realistic objects by using natural methods to alter geometry.

3. Fluid Simulation

In our animations, we use liquid to change the geometry of solid bodies, although we can also use gaseous fluids. We drive our fluids with a finite difference simulation, solving the incompressible Navier-Stokes equations for the velocity field at each time step and updating the fluid accordingly. To represent the free surfaces of liquids, we use particle level sets [EFFM02].

We create new scalar fields within our simulated fluid in order to track temperature, chemical concentration, and sediment. We manipulate these new fields with the environment of the simulation, and we diffuse and advect these fields along with the fluid momentum. Though diffusion is not vital for all fluid animations, it is important for some of our auxiliary scalar fields. Each simulation step, we advect and diffuse these scalar fields with the equation:

$$\frac{\partial S}{\partial t} = -(\mathbf{u}\cdot\nabla)S + \kappa\nabla^2 S \qquad (1)$$

where S is the scalar field, $\mathbf{u}$ is the velocity of the fluid, and κ is the diffusion coefficient. We assign a separate diffusion coefficient to each scalar field, because, for example, the thermal diffusion rate will typically differ from a chemical diffusion rate.

4. Geometry Modification

4.1. Representing the Solid Body

We use a level set stored in a regular 3-dimensional grid to model each solid object, similar to what Melek and Keyser [MK03] did to model burning objects. We will refer to this level set as the *solid level set*, in order to distinguish it from the particle level set associated with our fluid surface. We move the surface of the solid using the level set equation [OF03] [Set99]

$$\phi_t + \mathbf{V}\cdot\nabla\phi = 0 \qquad (2)$$

where $\mathbf{V}$ is the velocity field associated with the solid level set and ϕ is the implicit function that represents the solid surface. We use the convention that ϕ is zero at the surface of the solid, ϕ is positive outside the surface, and ϕ is negative inside (Figure 3). In order to alter the shape of our objects, we create a velocity $\mathbf{V}$ at the solid-fluid interface and advect the solid level set along this velocity. In most cases, we remove mass from our objects, so we create an inward velocity along the direction normal to the surface. We use the fast marching method [Set99] to maintain the solid level set's signed distance field in most of our examples. We refer the reader to [OF03] [Set99] for further details about implementation of level sets. We will show how to simulate erosion, sedimentation, and corrosion in order to calculate the velocity of this solid level set.

To aid our physical simulations, we use an additional density field that is co-located with the solid level set and contains scalar values that represent the local density throughout the object. This allows us to create inhomogeneous materials in which the density varies spatially. The density field is important for rigid body dynamics and density-dependent surface reactions. The per-object level set and density fields are distinct from the grid used for fluid simulation, and they don't even have to be on the same physical scale. To be more explicit, the motions of the rigid bodies and liquid are calculated at the resolution of the fluid grid, but the changing geometry of the rigid bodies is calculated at its own resolution.

The velocity field of the solid level set only encodes inward and outward velocities normal to the surface, for the purposes of simulating physical changes at the surface of

Figure 2: *Acid burning through a solid object and splitting it into two pieces.*

the object; we do not use this velocity field to translate or rotate the body. We use rigid body dynamics to reposition the solid level set grid in these instances, because it is more efficient, and it eliminates distracting artifacts like smoothing and mass-loss. For fast collision detection, we utilize the signed distance field that is maintained for our solid level set calculations.

For calculating erosion and corrosion, we utilize information from the fluid level set in order to assign surface velocities to the solid level set. To do this, we must transform the positions of the cells in the solid grid into the coordinates of the fluid grid, and then interpolate fluid velocity or scalar information there. We use a simple transformation matrix multiplication (translation, rotation, and scale) in order to transform from the solid level set coordinate system into the fluid coordinate system.

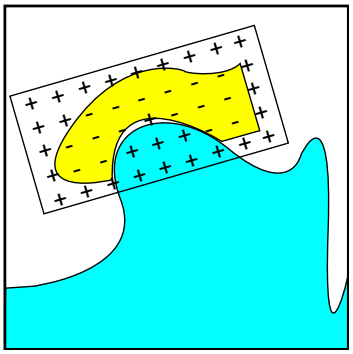

Figure 3: *The yellow region is the rigid body, and the blue region is liquid. The positive and negative values in the box represent the solid level set's signed distance field, which is in a different coordinate system than the fluid simulation. The liquid alters the solid level set to change the shape of the rigid body.*

4.2. Updating the Rigid Body

Because we change the geometry of our rigid bodies during the simulation, many of their physical properties need to be updated as well. We maintain an accurate account of each rigid body's volume, mass, density, center of mass, and moment of inertia in order for it to behave realistically. All of these properties will change as the geometry of the object changes.

We use the underlying solid level set and density field to update the physical properties. The volume of an object is found by iterating over the entire density field and accumulating a value for the amount of space that the field occupies. The signed distance function will be negative everywhere that the body exists, so we will simply increment the volume by $\Delta v = \Delta x \Delta y \Delta z$ for every negative signed distance function value in the solid level set. Similarly, we sum all values in the density field wherever the object exists in order to find the mass of the object.

$$m = \Delta v \sum_D \rho \tag{3}$$

where m is the mass of the object, ρ is the density function, and D is the interior of the rigid body.

The center of mass of the rigid body is found with the equation:

$$\mathbf{x} = \Delta v \frac{1}{m} \sum_D \mathbf{r}\rho \tag{4}$$

where $\mathbf{r}$ is the vector from some reference point to each element.

We calculate the rigid body's moment of inertia by summing over all positive density field elements:

$$I_{ij} = \Delta v \sum_D \left(\delta_{ij} \sum_k d_k^2 - d_i d_j \right) \rho \tag{5}$$

where I_{ij} represents an entry in the 3×3 inertia tensor, i is the row of the tensor, j is the column of the tensor, k is an iterator from 1 to 3, and δ_{ij} is the delta function. $\mathbf{d}$ is the position vector relative to the center of mass of the rigid body, where d_i is the ith component of the position vector.

Note that these discretizations are merely approximations to the physical properties of the rigid body. There are small errors associated with each parameter. For example, the volume calculation does not account for the fractional volume occupied near the surface of the object. However, assuming we maintain a density field that is more refined than the fluid grid, these errors are negligible when compared with the errors accumulated during solid-fluid coupling.

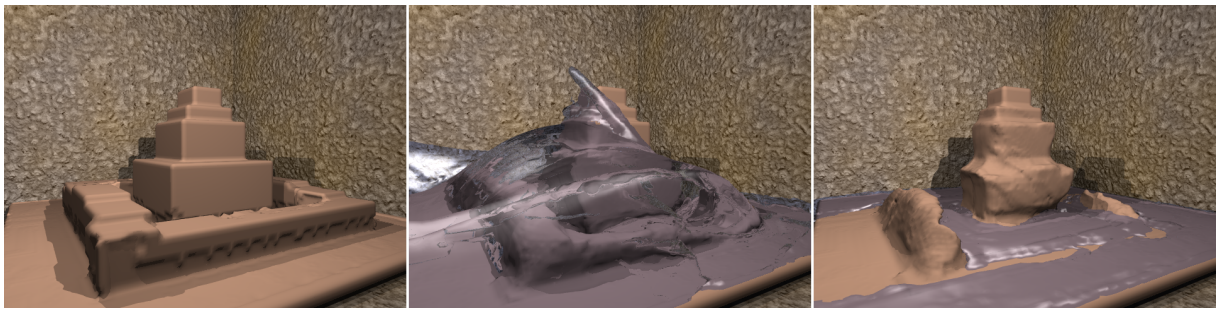

Figure 4: *Crashing waves erode a sand castle.*

4.3. Separation

As geometry changes throughout the animation, we occasionally encounter situations in which our solid objects split into multiple pieces. We must take this into consideration, otherwise detached fragments will hover near each other, acting with the same center of mass, velocity, and rotation. Our simulator has to recognize that these pieces are now separate rigid bodies, and no longer part of the same original body (Figure 2).

To accomplish this, we treat the negative values of our signed distance function as nodes in a graph with edges to each neighboring negative value. We then execute a connected component search in order to discover any broken pieces. We create a new rigid body for every new piece that is found, and we assign each new piece its own level set and density grid.

After the object splits, we need to calculate the resulting motion of each rigid body. The velocity of each new body is given by the equation:

$$\mathbf{v} = \mathbf{v}_0 + \omega \times \mathbf{d} \tag{6}$$

where $\mathbf{v}_0$ is the velocity of the original body, ω is the angular velocity before the separation, and $\mathbf{d}$ is the vector from the center of mass of the original body to the center of mass of the new body. The new angular velocity is exactly the same as the angular velocity before separation.

5. Simulating Natural Phenomena

The phenomena addressed in this paper are erosion, sedimentation, and corrosion due to strong acid. Each modifies the geometry of an object in its own way, and needs its own extra physical information.

5.1. Erosion

When water flows over a body made of rock, the particles in the water collide with particles comprising rock. With enough force, this friction can dislodge microscopic pieces of the rock and suspend them in the water. To simulate erosion, we must model how much of a solid will be swept away by a liquid with given physical properties.

Erosion occurs when a fluid flows parallel to the surface of an object. The parallel fluid forces create a displacement called a *shear* on the object, and the spatial derivative of the fluid's velocity is called the *shear rate*. The total stress that this shear rate exerts immediately inside the solid object is called the *shear stress*. To apply these ideas to a solid, we choose to give the solid object non-Newtonian fluid characteristics, via a power-law model:

$$\tau = K\theta^m, \tag{7}$$

where τ is the shear stress, K is a constant, θ is the shear rate and m is the power-law index, a constant determined by the material of the solid. With enough shear stress, the solid should deform like a liquid, so we treat our solid as a *pseudoplastic*, or *shear-thinning* fluid, because it "thins" with increasing shear. This is intuitive with a simple example: if you add shear stress to a pile of sand by kicking it, the sand will flow around your foot like a fluid. If you remove the shear, however, the sand will quickly settle and act as a solid mound. The power-law model is shear-thinning for $0 < m < 1$, with many shear-thinning fluids modeled by power-law indices on the order of $\frac{1}{2}$. We assume $m = \frac{1}{2}$ for our erosion simulations.

We determine the erosion rate using the equation formulated by Partheniades [Par65]:

$$\varepsilon = k(\tau - \tau_c)^a, \tag{8}$$

where ε is the erosion rate, k is the erosion constant, τ is the shear stress experienced by the solid due to the fluid's motion, τ_c is the critical shear stress of the solid, and a is another constant. The value of a is often assumed to be 1, and we are free to assign an arbitrary value to k. τ_c is the critical shear stress that needs to be overcome in order for erosion to take place. In other words, erosion will only occur with sufficiently high shear stress from the fluid.

To determine the shear stress by equation 7, we simply approximate the shear rate θ from the velocity of the fluid relative to the solid surface, $\mathbf{v}_{rel}$, and a small constant distance l over which the shear is applied:

$$\theta = \frac{\mathbf{v}_{rel}}{l}. \tag{9}$$

In our implementation, we set l to be the width of one cell in the fluid grid. We retrieve the velocity of the liquid from the velocity of the fluid simulation at a distance l away from the solid surface, and we recover the velocity of the solid surface using equation 6, where $\mathbf{d}$ is a vector from the center of mass to the surface. Once we know the shear rate and shear stress, we can compute the erosion rate at the surface of the solid using equation 8. We will use this erosion rate as the velocity of the solid level set interface. To add an element of user control to our erosion simulation, we implemented a "critical shear stress map." Areas of the map with a higher τ_c generally maintain their shape better than other regions. We used an erosion map in figure 4 to allow waves to erode only the sand castle, not the beach beneath it.

5.2. Sedimentation

In order to include effects caused by sedimentation of suspended matter, we create an additional scalar field to track the local mass density of suspended particles. These suspended solids could be transported into the field of simulation by, e.g., an inflowing stream, or could be a result of local erosion processes. In the event that the suspended material originates from erosion in the field of simulation, we simply supplement the erosion step (described above) with a step where we deposit all of the removed mass directly into the suspended mass field. In other words, we remove an amount of mass from a solid region and add that same quantity into the suspended mass field in the neighboring fluid region.

To determine the quantity of mass to suspend in the fluid, we use the velocity of the solid interface and the time step in an Euler approximation. This is only first order, but it does not produce any distracting visual artifacts. If desired, one can implement more accurate volume-conserving schemes.

We consider suspended particles that are denser than water, so they will slowly sink and settle on the ground. We model this settling by a vertical *settling velocity* relative to the local fluid motion,

$$\mathbf{u}_{settling} = \frac{2}{9} r^2 \frac{\rho_s - \rho_f}{\mu} \mathbf{g} f(c), \tag{10}$$

with the settling velocity $u_{settling}$ dependent on the particle radius r, the solid particle density ρ_s, the fluid density ρ_f, the fluid viscosity μ, and the gravity vector $\mathbf{g}$. The *hindered settling function* $f(c)$ is based on the local solid volume fraction c, the percentage of local volume occupied by particles. The Richardson-Zaki relation [RZ54], $f(c) = [1 - (c/c_{max})^n]$, is an accepted empirical approximation, with c_{max} a maximum solid volume fraction and n an exponent typically 4–5.5. We choose $c_{max} = 0.7$ and $n = 5$ here. At vanishingly small c, this model agrees with the low Reynolds number Stokes settling velocity of spherical particles [Bat70].

We use the same particle field in our sedimentation simulations that we used for our erosion calculations, because solid material in the fluid is necessarily suspended. We advect the suspended particle field with the velocity of the surrounding liquid plus the settling velocity (equation 10), so the suspended material settles out as the simulation runs. We assume diffusion is minimal, and we use the solid material field value in a given volume cell to determine the volume fraction c.

In order to deposit sediment onto a ground surface, we treat the ground as a solid body with a solid level set representing its geometry. We assume that any sediment advected though the surface of the rigid body should be deposited, and we will use this as the trigger for our algorithm. We seek to expand the surface of the solid, because we are adding mass here, unlike in our erosion steps. Therefore, instead of adding an inward velocity to the solid interface, we will add a velocity in the direction opposite of gravity wherever sediment should build up. The rate at which mass builds up on the surface is determined by the speed at which mass is advected through the solid surface, multiplied by the ratio of the densities of the suspended particles to the density of the solid.

$$u_{interface} = -u_{settling} \frac{\rho_{suspended}}{\rho_{ground}} \tag{11}$$

It is possible to run both erosion and sedimentation in the same simulation, in order to simulate the transport of soil through water (Figure 5). This is fairly simple, as both simulations share the same scalar field.

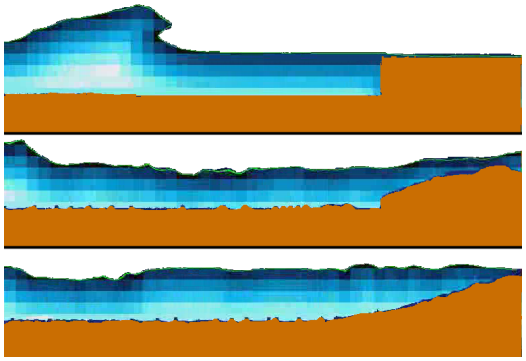

Figure 5: *Combining erosion and sedimentation produces a realistic shore-sculpting effect*

5.3. Strong Acid

Acids combine with metals in an exothermic reaction, yielding hydrogen gas and an additional compound. For simplicity, we choose to simulate only first-order reactions with strong acids. Strong acids completely dissociate in water and the chemical reaction never reverses, so the equations are easy to implement. To demonstrate a well-understood chemical reaction, we will use hydrochloric acid, HCl, as our strong acid, and magnesium, Mg, for our metal. The

balanced equation for the reaction between these two substances is:

$$2\mathrm{HCl}(aq) + \mathrm{Mg}(s) \Longrightarrow \mathrm{MgCl}_2(aq) + \mathrm{H}_2(g) + q \qquad (12)$$

where HCl is the hydrochloric acid, Mg is the magnesium, $MgCl_2$ is the additional product, H_2 is hydrogen gas, and q is heat expelled by the reaction. (aq) indicates that the chemical is dissolved in water (aqueous solution), (s) indicates that the chemical is a solid, and (g) indicates a gaseous chemical.

This reaction basically summarizes the transformation of a solid piece of metal into a gas. The solid magnesium changes into dissolved $MgCl_2$, and the hydrogen from the acid turns into gas. Because water does not explicitly enter into this reaction, we assume that the volume of the water remains constant. We do not simulate the expelled gas for our animations, but a simulation of fumes or buoyant bubbles rising from the reaction site might exaggerate the effect of the acid to make the animation more interesting.

The reaction rate is defined as the number of chemical reactions per unit time. This reaction is first order in terms of the concentration of the acid, or

$$r = k[HCl] \qquad (13)$$

where r is the reaction rate in units of reactions per unit time, and k is the rate constant. The rate constant can be determined using the Arrhenius equation, $k = Ae^{[-E_a/(RT)]}$, where A is the Arrhenius constant, E_a is the activation energy, R is the gas constant, and T is the temperature. The Arrhenius constant is an experimentally determined value, and we will assume the activation energy is constant throughout the reaction. We can assign arbitrary values to these variables in order to tune our animations. Consequently, some of the constants in the Arrhenius equation can be collapsed:

$$k = Ae^{-B/T}. \qquad (14)$$

Combining the equations for rate with equation 12, we get:

$$\frac{d[\mathrm{Mg}]}{dt} = -Ae^{-B/T}[\mathrm{HCl}] \qquad (15)$$

and a further discretization gives us:

$$\Delta[\mathrm{HCl}] = -2Ae^{-B/T}[\mathrm{HCl}]\Delta t \qquad (16)$$

$$\Delta q = Ae^{-B/T}[\mathrm{HCl}]\Delta t \qquad (17)$$

where [HCl] is the concentration of hydrochloric acid, and [Mg] is the concentration of magnesium. Because this concentration is in units of metal per unit volume, the concentration of magnesium is equivalent to the density of the metal. From this point forward, we will use the concentration of metal and the density of the metal interchangeably. These equations tell us exactly how to change the acid concentration, how much heat to release, and how much to modify the density at the surface of the metal at each time step.

Note that this reaction rate is not specific to acid and metal. Many chemical reactions can be simulated with our method. For example, reactions between sodium and water, or dissolving effervescent baking soda tablets (Figure 1) obey similar chemical equations.

To modify the geometry with this chemistry simulation, we use extra scalar fields for acid concentration and temperature. An optional scalar field can represent the hydrogen gas fumes or the soluble compounds produced by the reaction. We do not need an extra field for the concentration of the metal because we will use the metal's underlying density field for this purpose.

The surface of the metal will change with the velocity $\mathbf{u}_{interface} = -(d[\mathrm{Mg}]/dt)\mathbf{n}$, where $\mathbf{n}$ is the surface normal, and $d[Mg]/dt$ is the rate defined in equation 15. We advect the solid level set along this interface velocity in order to deform the body. We also change the values in the acid concentration field using equation 16 and feed more heat into the temperature field using equation 17 during this step.

The reaction rate increases with increasing heat, and the heat production is linear in terms of the reaction rate. This can produce quite a range of effects. If thermal diffusion is large, the reaction will mostly depend on the acid concentration. If thermal diffusion is minimal and B in equations 16-17 is large, the reaction rate is highly dependent on temperature. This can cause reactions to accelerate where the metal has more surface area (Figure 6).

Figure 6: *This 2-dimensional simulation displays a "pitting" effect caused by a high temperature gradient.*

6. Interactive Application

If we simplify our implementation by speeding up or removing the fluid simulation, we can perform our geometry modification at interactive rates. In our particular application, the user can spray acidic particles into a scene and cause the geometry to change shape using our corrosion model. Figure 7 shows how we can use our technique to burn through a wall in real time. Simulations like this may be useful in video games, for example, where the hero must dissolve a villian character or escape from a prison cell by procedurally destroying it.

To create this application, we placed a solid level set object into a 3d interactive world, and we allowed the user to shoot balls of acid at it. When a ball collides with the surface of the solid level set, we delete the ball and deposit acid onto a chemical grid co-located with the solid level set at the

point where the ball collided. We use the chemical grid and equation 15 to calculate the inward velocity of the solid level set. We then advect both the solid level set and the chemical concentrations along this velocity. Finally, we use marching cubes to extract the surface of the solid level set at the end of each simulation step.

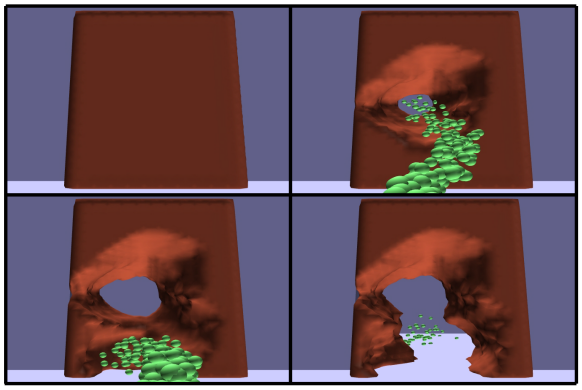

Figure 7: *Using our interactive game, the user shoots a stream of acidic particles at a metal obstacle. The metal changes shape due to corrosion at interactive rates using our algorithm.*

7. Results and Discussion

We implemented our corrosion simulator to animate a reaction between acid and baking soda in Figure 1. The bubbles rising from the reaction site are small particles that we passively advect with a buoyant force (similar to Greenwood and House [GH04]), and the number of bubbles is proportional to the amount of gas produced by the chemical reaction. If we wanted additional realism at the expense of more computational cost, we could have simulated bubbles with a two-phase flow like [HK03], [HK05], or [KLL*07].

In another animation, we pour acid onto a metal block that is suspended across two pillars (Figure 2). The center of this block eventually corrodes away, and the metal splits into two pieces. The pieces then fall and rotate independently. This demonstrates the treatment of separation that was presented in section 4.3. We use one-way solid-fluid coupling to animate the falling blocks in this example; the block can push the fluid, but the fluid cannot push the block.

We illustrate our erosion techniques by destroying a sand castle with waves (Figure 4). The waves flow around the central tower and wash away the side walls. The water with high velocity removes a significant amount of structure from the castle, and the sediment is swept out to sea. We use a critical shear stress map to ensure that the beach beneath the castle is not eroded. Note that water can wash away materials to create overhanging sand in our erosion model, which is not possible using height field erosion [MKM89]. We also apply erosion to a cow statue in Figure 8. The erosion rate is highest at the beginning, when the liquid's velocity peaks. The wave immediately tears the legs from that statue and later smooths out fine details like the horns and ears.

Our off-line fluid simulations run at reasonable speeds, with times ranging from a few hours to almost a day. The animations typically spend about five to ten percent of the computation time with our solid level sets. Our most expensive animation was the sand castle in Figure 4. We ran a $120 \times 60 \times 100$ fluid simulation with the $160 \times 80 \times 160$ solid level set model of the sand castle. The entire simulation ran for 18.4 hours, and 7% of this time was spent on our solid level set. Even when the resolution of our solid level set is more than double the resolution of the fluid level set, the simulation time is still dominated by the fluid dynamics calculations.

By removing the fluid simulation, as in section 6 and figure 7, we achieve interactive rates. We implemented this algorithm entirely in software with un-optimized research code, and we achieved rates of 5 frames per second with a $20 \times 30 \times 44$ simulation grid. We speculate that we will achieve real time performance with far more detailed simulations once we implement this on the newest generation of graphics hardware.

8. Conclusion and Future Work

We have presented techniques for animating a number of natural phenomena, namely erosion, sedimentation, and

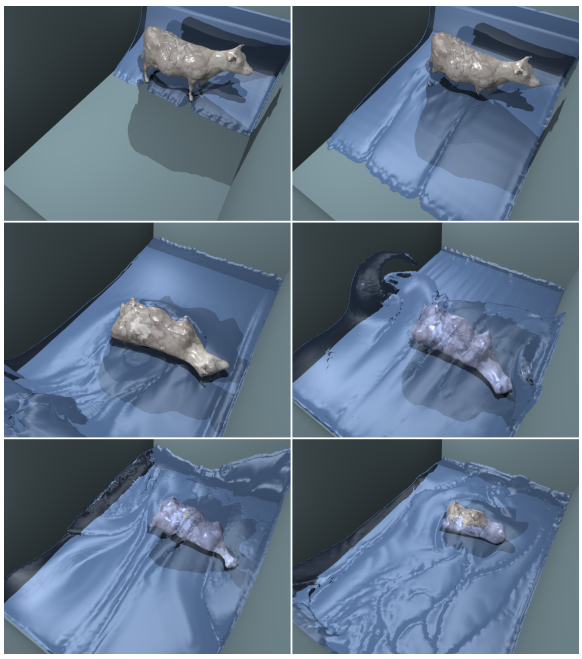

Figure 8: *A rush of water erodes a cow statue.*

acidic corrosion. We acknowledge that our simulations may appear synthetic and unnatural, and we attribute that to the raw complexity of the phenomena of erosion and corrosion. We believe that some of this unnaturalness could be alleviated with slower but more detailed simulation methods. For example, more accurate bubbling effects could be seen if we used two-phase flow like Hong *et al.* [HK05], and more detailed pitting could result if we animated our acid as tiny droplets like Wang *et al.* [WMT05]. The sandcastle example could be more realistic if we incorporated flowing sand as in [ZB05] or granular sand as in [BYM05]. If we implemented physically based fracturing, we could animate secondary effects of erosion, like the crumbling and collapsing of weakened structures.

9. Acknowledgements

The authors would like to thank R. Brooks Van Horn III for helping us render our animations, Spencer Reynolds for creating the videos, and Nathan Sisterson for creating lovely sound effects. We would also like to thank the members of the CMU and Georgia Tech graphics group, in particular Huamin Wang and James Vanderhyde.

References

[Bat70] BATCHELOR G. K.: *An Introduction to Fluid Dynamics*. Cambridge University Press, Cambridge, 1970.

[BBB07] BATTY C., BERTAILS F., BRIDSON R.: A fast variational framework for accurate solid-fluid coupling. In *SIGGRAPH* (2007).

[BTHB06] BENES B., TESINSKY V., HORNYS J., BHATIA S.: Hydraulic erosion. In *Computer Animation and Virtual Worlds* (2006), pp. 1–10.

[BYM05] BELL N., YU Y., MUCHA P. J.: Particle-based simulation of granular materials. In *ACM SIGGRAPH/Eurographics symposium on Computer animation* (New York, NY, USA, 2005), ACM Press, pp. 77–86.

[CDM*02] CUTLER B., DORSEY J., MCMILLAN L., MUELLER M., JAGNOW R.: A procedural approach to authoring solid models. In *SIGGRAPH* (2002), pp. 302–311.

[CGFO06] CHENTANEZ N., GOKTEKIN T., FELDMAN B., O'BRIEN J.: Simultaneous coupling of fluids and deformable bodies. In *ACM SIGGRAPH/Eurographics symposium on Computer animation* (2006), pp. 83–89.

[CMT04] CARLSON M., MUCHA P. J., TURK G.: Rigid fluid: Animating the interplay between rigid bodies and fluid. In *SIGGRAPH* (2004).

[CS03] CHANG Y.-X., SHIH Z.-C.: The synthesis of rust in seawater. *The Visual Computer 19* (2003), 50–66.

[DEL*99] DORSEY J., EDELMAN A., LEGAKIS J., JENSEN H. W., PEDERSEN H. K.: Modeling and rendering of weathered stone. In *SIGGRAPH* (1999), pp. 225–234.

[EFFM02] ENRIGHT D., FEDKIW R., FERZIGER J., MITCHELL I.: A hybrid particle level set method for improved interface capturing. *J. Comput. Phys. 183*, 1 (2002), 83–116.

[GH04] GREENWOOD S. T., HOUSE D. H.: Better with bubbles: enhancing the visual realism of simulated fluid. In *ACM SIGGRAPH/Eurographics symposium on Computer animation* (New York, NY, USA, 2004), ACM Press, pp. 287–296.

[GSLF05] GUENDELMAN E., SELLE A., LOSASSO F., FEDKIW R.: Coupling water and smoke to thin deformable and rigid shells. In *SIGGRAPH* (2005), pp. 973–981.

[HK03] HONG J.-M., KIM C.-H.: Animation of bubbles in liquid . *Comput. Graph. Forum 22*, 3 (2003), 253–262.

[HK05] HONG J.-M., KIM C.-H.: Discontinuous fluids. In *SIGGRAPH* (2005), pp. 915–920.

[IKC04] IHM I., KANG B., CHA D.: Animation of reactive gaseous fluids through chemical kinetics. In *SIGGRAPH / EUROGRAPHICS Symposium on Computer Animation* (2004), pp. 203–212.

[KLL*07] KIM B., LIU Y., LLAMAS I., JIAO X., ROSSIGNAC J.: Simulation of bubbles in foam by volume control. In *SIGGRAPH* (2007).

[LIGF06] LOSASSO F., IRVING G., GUENDELMAN E., FEDKIW R.: Melting and burning solids into liquids and gases. *IEEE TVCG 12* (2006), 343–352.

[McC91] MCCANN M.: *LIGHTS! CAMERA! SAFETY! A Health and Safety Manual for Motion Picture and Television Production*. The Center for Safety in the Arts, 1991.

[MDG01] MERILLOU S., DISCHLER J.-M., GHAZANFARPOUR D.: Corrosion: Simulating and rendering. In *Proceedings of Graphics Interface* (2001), pp. 167–174.

[MK03] MELEK Z., KEYSER J.: Interactive simulation of burning objects. In *Pacific Graphics* (2003), pp. 462–466.

[MKM89] MUSGRAVE F. K., KOLB C. E., MACE R. S.: The synthesis and rendering of eroded fractal terrains. *Computer Graphics 23*, 3 (1989), 41–50.

[OF03] OSHER S., FEDKIW R. P.: *Level Set Methods and Dynamic Implicit Surfaces*. No. 153 in Applied Mathematical Sciences. Springer-Verlag, New York, 2003.

[Par65] PARTHENIADES E.: Erosion and deposition of cohesive soils. *Journal of Hydraulics Division of the American Society of Agricultural Engineers 91* (1965), 105–139.

[RZ54] RICHARDSON J. F., ZAKI W. N.: Sedimentation and fluidization i. *Transactions on the Institution of Chemical Engineers 32* (1954), 35–53.

[Set99] SETHIAN J. A.: *Level Set Methods and Fast Marching Methods*. Cambridge University Press, 1999.

[WMT05] WANG H., MUCHA P. J., TURK G.: Water drops on surfaces. In *SIGGRAPH* (2005), pp. 921–929.

[ZB05] ZHU Y., BRIDSON R.: Animating sand as a fluid. In *SIGGRAPH* (2005), pp. 965–972.