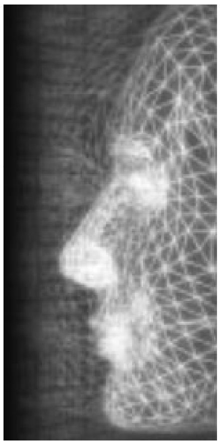
COMPUTER ANIMATION AND VIRTUAL WORLDS
Comp. Anim. Virtual Worlds (in press)
Published online in Wiley InterScience (www.interscience.wiley.com). DOI: 10.1002/cav.77

Hydraulic erosion



By Bedřich Beneš, Václav Těšínský, Jan Hornyš and Sanjiv K. Bhatia*

This paper presents a generalized solution to modelling hydraulic erosion using ideas from fluid mechanics. The model is based on the Navier–Stokes equations, which provide the dynamics of velocity and pressure. These equations form the basis for the model to balance erosion and deposition that determine changes in the layers between water and erosion material. The eroded material is captured and relocated by water according to a material transport equation. The resulting model is fully 3D and is able to simulate a variety of phenomena including river meanders, low hill sediment wash, natural water springs and receding waterfalls. The simulations show the terrain morphogenesis and can be used for animations as well as for static scene generation. Copyright © 2006 John Wiley & Sons, Ltd.

Received: 8 December 2004; Accepted: 18 May 2005

KEY WORDS: hydraulic erosion; virtual world; visualization; terrain morphing

Introduction

Erosion is the most important phenomenon in nature that guides the shape of a terrain over time. It results in the relocation of material on a surface, resulting in a modification of terrain relief. Erosion can occur due to a variety of agents. For example, running water forms valleys and ridges, material deposition results in sandstones and wind creates wind ripples. It is an inherently complex process, making it a hard problem for simulation.

In this paper, we limit ourselves to physically based simulation of hydraulic erosion. The model is based on the Navier–Stokes equations, the model of material transportation in a liquid, and the erosion–deposition model. Our technique simulates a wide variety of phenomena, provides visually plausible results in reasonable computational time and allows a certain level of control by the user. The technique can be used in a variety of situations to provide static terrain models as well as animations of terrain morphology.

Terrain morphs over time by both constructive as well as degenerative processes. Constructive processes result in features such as mountains, volcanoes and tectonic fractures. Degenerative processes, including erosion, are more important in shaping the terrain over a much longer period, and result in features such as gorges and canyons.

Erosion can be described as a three-step process. In the first step, the regolith or boundary between two surfaces is damaged. Different factors allow us to classify this process as water or hydraulic erosion, wind erosion, thermal erosion or erosion due to biological influences. In the second step, material is transported by natural factors such as gravity, water and wind. The material can also be transported by human influence but we ignore that in our model as we concentrate on modelling just the natural phenomena. The last step involves deposition of material.

We present an example of erosion simulation in Figure 1. The first image in the sequence shows the water entering in the water system, the second shows the water acting on the surroundings, and the last image shows the change in relief. We show deposited material in a different colour. We have displayed this system with a photon mapper, which clearly shows the caustics on the river bottom and the walls.

In this paper, we introduce a new algorithm for the simulation of hydraulic erosion phenomena. We model terrain erosion and deposition using velocity and pressure fields as given by the solution of Navier–Stokes equations. The algorithm involves transportation of material captured by water to a new location. The technique is physically based and allows the simulation of various phenomena in addition to hydraulic erosion.

This paper is organized as follows. In the next section, we present existing work on simulation of erosion from

*Correspondence to: B. Beneš, Department of Computer Graphics Technology, 1419 Knoy Hall, Room 363, Purdue University, West Lafayette, IN 47907, USA.
E-mail: bbenes@purdue.edu

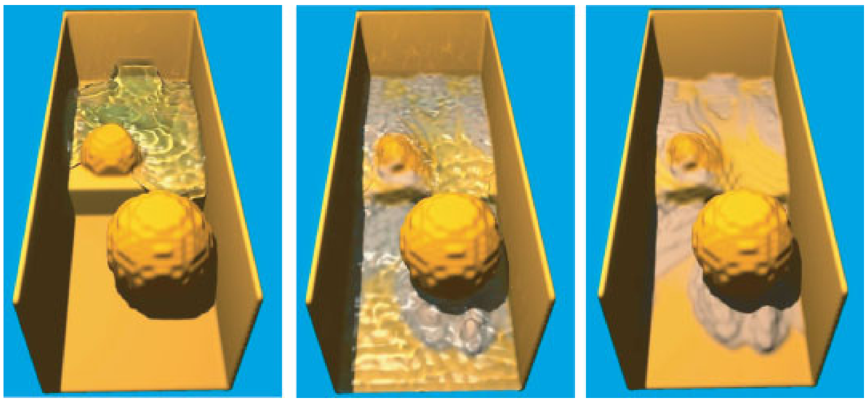


Figure 1. Water moving through a system.

the point of view of computer graphics. This is followed by our algorithm for hydraulic erosion. Thereafter, we provide the application and implementation details. The paper concludes with experimental results and future work.

Simulation of Erosion in Computer Graphics

Simulation of erosion has been an active area of research in computer graphics. In one of the earlier articles on the subject, Kelley *et al.*[1] have explored the use of fractals to create three-dimensional models of terrain with realistic appearance. They have provided a useful section on geological terminology and create topographical features by tracking a stream network, using fractals to provide variable degree of detail. They have used empirical erosion models based on simulation of drainage systems and their model results in simplistic hill-slopes.

Another fractal-based model was presented by Musgrave *et al.*[2] They termed their model *noise synthesis* to emphasize locally independent control of fractal dimensions and surface characteristics. This model is usable for both thermal and hydraulic erosion. The thermal erosion is based on a low-pass filter to smooth the terrain. Hydraulic erosion is based on material capture and relocation using simple and 'ad hoc' gradient-based diffusion for transport. Material transport within water is ignored for the model.

Chiba *et al.*[3] presented a 'quasi-physically based' model that uses velocity fields of water flow to cause erosion. This method results in a simulation of eroded topography of natural mountains. Their goal was to produce large-scale ridge and valley lines to reflect the water flow. The method involves simulation of motion of water particles along mountain faces in discrete time steps to show erosion, transportation and sedimentation.

Another interesting physically based technique for modelling erosion shows the weather effects on stones over time.[4] The technique is based on a data structure called *slab* that creates a layer of voxels for simulation. The simulation is confined to *chemical weathering* of stones using the solution of Navier–Stokes equations to simulate the transportation of sediment within the stone structures, by integrating the effects across slabs.

Beneš and Forsbach[5] have described a hydraulic erosion–deposition model. In this model, water dissolves sediment, causing it to be transported from one point to another. Water evaporates, leaving behind the sedimentation. This model utilizes diffusion to simulate water and material transportation.

Li and Moshell[6] described a physically based model of slippage of mud that has been applied to a real-time manipulation digging machinery. Along similar lines, Sumner *et al.*[7] introduced an algorithm to simulate the effects of object motion in soft ground, such as sand, snow or mud. Their simulation model uses a regular height field defined by vertical columns of material and is based on five independent parameters—roughness, liquidity, compression and two parameters for slope—to control material properties. The surface deformation is simulated by compression or displacement of material under a rigid geometric object by shifting columns in the height field grid and then smoothing the grid.

Onoue and Nishita[8,9] have proposed an efficient algorithm to form wind ripples in a desert. In their model, the particles of sand are affected by wind and shift by *saltation* (sand particles jump up due to wind) or *creep* (sand particles move along the surface without jumping up). Reference 8 describes a simple algorithm resulting in moving wind ripples. In reference 9 they presented an interactive application, called *virtual sandbox*, that allows manipulations with a model of sand. In the virtual sandbox, sand can be represented as a regular-height field or as particles. When it is not elevated, the sand is smoothed by a thermal erosion algorithm. When it is elevated, the sand is relocated by using a particle representation.

All of the work mentioned so far has concentrated on simulation and rendering, with an emphasis on realistic display. We want to mention the work of Langendoen[10] in the field of Geographical Information Systems (GIS) as the technique presented in this paper is an extension of one of his models.

A typical representation used in GIS modelling is in one dimension. For example, a river is not supposed to change its trajectory and can be modelled by a function in one dimension to describe the material transport. This function describes only the depth of the river at different places. In contrast, a 2D model typically uses a discrete representation of the terrain. The most common representations are either a regular mesh of squares or an irregular network of triangles. This allows for the calculation of erosion in two independent steps: (1) for the river bed, and (2) for the river banks. The CONCEPTS paradigm[10] provides a physical model of the qualitative description of erosion in the river bed and the river bank. This model serves as a basis for our techniques.

The main disadvantage of most of the techniques mentioned in this section is that they are applicable to only a limited set of erosion models. In addition, the majority of these techniques rely on regular height fields for modelling which may not allow formation of structures in three dimensions. The model presented in this paper works with voxels and allows for simulation of various phenomena such as receding waterfalls, meanders and springs of water. We shall provide the basic erosion algorithm and model in the next section.

Erosion Algorithm

The erosion algorithm presented in this section combines elements of Navier–Stokes equations, the erosion–deposition model and the model of material transportation. The algorithm requires a model of the environment as a regular voxel grid. It defines voxels corresponding to liquid, air and material objects and classifies those voxels as inflow (source) and outflow (drain) voxels. After the set-up is performed, the algorithm operates in discrete time step iterations.

In each iteration, the algorithm solves the Navier–Stokes equations to compute the pressure and the velocity field in each voxel. Then, it applies the erosion–deposition on water-material boundaries. The last step in the iteration involves transporting the material through drop, diffusion and velocity.

Fluid Dynamics

Fluid dynamics describes the movement and interaction of liquids, gases and solids in a system. We are primarily interested in the movement of liquids, and hence can use Navier–Stokes equations to provide a model that describes the movement of incompressible liquid at an approximately constant temperature and density. Given the velocity field of a fluid $\boldsymbol{u}$ and pressure p for an initial time instant $t = t_0$, the development of $\boldsymbol{u}$ and p over time is completely given by the Navier–Stokes equations as follows:

$$\nabla \cdot \boldsymbol{u} = 0 \tag{1}$$

$$\frac{\partial \boldsymbol{u}}{\partial t} = -(\boldsymbol{u} \cdot \nabla)\boldsymbol{u} - \frac{1}{\rho}\nabla p + \nu \nabla^2 \boldsymbol{u} + \boldsymbol{f} \tag{2}$$

where ρ is the liquid density, ν is the liquid viscosity and $\boldsymbol{f}$ is an external force. Equation (1) reflects the incompressibility and mass conservation of liquid. Equation (2) expresses the conservation of momentum by combining the pressure field and liquid velocity.

There are a number of solvers for the above equations that focus on different areas of interest. The computational fluid dynamics (CFD) methods for modelling liquid flow prefer precision. However, they are computationally expensive, making them unsuitable for large-scale simulations in computer graphics. Foster, in two separate articles with Metaxas[11] and Fedkiw,[12] introduced a practical approach to model and animate the flow of liquids that has been widely accepted in computer graphics. They provided a solution through a finite difference approximation that works on a low-resolution regular 3D mesh of voxels and simulated the level by marking particles. This technique does not provide a precise solution but it is suitable for application in computer graphics.

The solution gives the velocity field in each face of a voxel, calculating the pressure p at its centre. It then uses the contents of the voxel to classify it into one of the following three classes:

1. FULL—Voxel is full of water. The voxel may or may not contain any material. If it does contain material, the material may be dissolved in liquid, up to a saturation point as explained in detail in the subsection on state changes.
2. EMPTY—Voxel does not contain anything except air. This is used for simplification in this model though we can extend it by adding material into this voxel type.
3. MAT—Voxel contains some material.

The model pays special attention to boundary voxels. The pressure and velocity vectors for a voxel depend on the viscosity of the material and on the liquid velocity in the adjacent voxels.

Erosion–Deposition and Transport

Erosion affects the landscape on a macro level but the interesting phenomena, from the visual point of view, occur at a relatively small scale. For example, rain and floods affect a large area of land but rivers cover a relatively small part of the overall landscape. This also implies that there is a sort of focus that needs to be addressed in modelling erosion. It is difficult and computationally expensive to perform erosion simulation over a large landscape, and therefore we restrict ourselves to high-precision simulation of erosion in a smaller focus area of the landscape. We will perform the simulation by treating the system as half-opened. We will describe the highly exposed areas, and a set of inflows and outflows.

Foster and Metaxas[11] simulated the flow of liquid by keeping a constant quantity of material in a voxel. In our algorithm, we define a scalar field variable m that describes the quantity of material inside a FULL or MAT voxel. In the case of a FULL voxel, m describes the amount of dissolved material. In the case of a MAT voxel, m describes the amount of material contained in the voxel. In theory, the variable m can also be associated with an EMPTY voxel to describe the amount of material transported by wind.

State Changes

The MAT voxels with $0 < m < 1$ are located on the boundary between water and material, or air and material, voxels. A water voxel cannot carry more sediment than its saturation, denoted by $c_{\max}$. Therefore, each FULL voxel must satisfy the condition $0 \leq m < c_{\max}$. Similarly, a MAT voxel that is not on a boundary has $m = 1$.

A voxel can change its state by either of the following two transitions: FULL $\Rightarrow$ MAT to describe deposition of material and MAT $\Rightarrow$ FULL to show erosion. Deposition always occurs towards the bottom of the voxel, which provides an important cue for rendering. A FULL voxel remains in its state as long as the amount of material is smaller than its saturation capacity. When its actual capacity m becomes more than the saturation $c_{\max}$, it changes state by the transition FULL $\Rightarrow$ MAT. The excess material, given by $m - c_{\max}$, is distributed to all the adjacent voxels that are FULL. The MAT voxel retains its state as long as $m \neq 0$.

Mathematical Model

Our erosion–deposition model is based on the model presented by Langendoen[10] using the sediment transportation equation. Let C represent the concentration of the sediment mass, E represent the entrainment rate of the material (erosion), and D represent the deposition rate. Then, the sediment transportation equation is given by

$$\frac{\partial C}{\partial t} + (\boldsymbol{u} \cdot \nabla)C = E - D \tag{3}$$

It should be noted that E and D have different values for a cohesive or cohesionless material.

Cohesive Material. The cohesive bed-material has a form where the entrainment rate E and deposition rate D are described by

$$\begin{aligned} E &= e\left(\frac{\tau}{\tau_{ce}} - 1\right) \\ D &= \omega d\left(1 - \frac{\tau}{\tau_{cd}}\right) \end{aligned} \tag{4}$$

Here e represents the erosion-rate constant ($e = 0.01$), τ is the bed shear stress and is a function of the geometry of the object. τ_{ce} is the shear stress strength of the material. In the second part of equation (4), d is the deposition-rate constant ($d = 0.01$), τ_{cd} denotes the shear stress strength of the material below which the particles start to deposit, and ω is the fall velocity within the liquid ($\omega = 0.2$). The term $1 - \tau/\tau_{cd}$ is the probability that the particle will stick with the material and not re-enter into the flow. Clearly, if $\tau > \tau_{cd}$, D is set to zero. In our experiments, we have used the values $\tau_{ce} = \tau_{cd} = 10$. If $f = 1/2$ denotes the material friction and $\rho = 0.001$, the value of τ is given in terms of viscosity v and density ρ by

$$\tau = \frac{f\rho v^2}{2} \tag{5}$$

Equation (3) also reflects the material transport that is described implicitly by the Navier–Stokes equations in our model. Hence, the numerical solution ignores the transportation and solves

$$\frac{\mathrm{d}C}{\mathrm{d}t} = E - D \tag{6}$$

Solving equation (6) by Euler's method for each time interval Δt, we have

$$C^{n+1} = C^* + \Delta t(E^* - D^*) \tag{7}$$

E^* and D^* are obtained as

$$\begin{aligned} E^* &= e\left(\frac{\frac{1}{2}\left(\tau^{n+1}+\tau^n\right)}{\tau_{ce}} - 1\right) \\ D^* &= \omega d\left(1 - \frac{\frac{1}{2}\left(\tau^{n+1}+\tau^n\right)}{\tau_{cd}}\right) \end{aligned} \tag{8}$$

Cohesionless Material. In the case of cohesionless material, we have to make a distinction between the equilibrium sediment mass, denoted by $\hat{C}$, and the actual concentration of material, denoted by C. If T provides the time-scale representing the adjustment rate of sediment mass from C to $\hat{C}$, the non-equilibrium adjustment of cohesionless bed-transport is given by

$$E - D = \frac{1}{T}(\hat{C} - C) \tag{9}$$

There are various definitions of T in reference 10 that describe different materials involving more parameters. We have chosen to define T as a material constant for our model.

The equilibrium sediment mass $\hat{C}$ depends on the local instantaneous flow. We represent it by the expression $\hat{C} = \min(k\|\boldsymbol{u}\|, c_{\max})$ where k is a constant (we have used $k = 0.7$). This approach relates the velocity of the fluid with the saturation $c_{\max}$ so that the deposition is higher in areas with low fluid velocity and there is an increase in erosion in areas with high fluid velocity.

We rewrite equations (3) and (9) as

$$\frac{\partial C}{\partial t} + (\boldsymbol{u} \cdot \nabla)C = q \tag{10}$$

$$\frac{\mathrm{d}C}{\mathrm{d}t} = \frac{1}{T}(\hat{C} - C) \tag{11}$$

Equation 10 gives C^* and is solved by the method of characteristics.[13] Its solution (C^*) is used as the initial value problem for equation (11), which has a known solution in the analytical form as

$$C^{n+1} = \hat{C} + (C^* - \hat{C})e^{-\frac{\Delta t}{T}} \tag{12}$$

Thus, C^{n+1} can be simply computed from C^* and $\hat{C}$.

Implementation Details

Our implementation proceeds in two phases: the simulation of erosion phase and the visualization phase. The erosion simulator uses OpenGL preview, which allows the user to stop the simulation when it does not proceed in the desired manner. It then uses a raytracer to render photorealistic scenes. Also, it saves the results of simulation for subsequent use.

The first phase in simulation—the simulation of erosion—has three parts: solving the Navier–Stokes equations, creating a numerical model of material transport, and creating the erosion–deposition model.

The Navier–Stokes equations solver starts with reading the initial conditions. For each time interval Δt, we perform the following steps:

1. We solve fluid dynamics problem using the Navier-Stokes equations solver. This gives us the pressure field p and the velocity field $\mathbf{u}$.
2. This step depends on the modelling of cohesive or cohesionless material:
   - For the *cohesive model*, the erosion and deposition are calculated as follows. First, we find all the voxels on the boundary of water and material. Then, we apply equation (4) to compute erosion and transfer the corresponding amount of material from the `MAT` voxels to the `FULL` voxels. Then, we calculate the deposition from the `MAT` voxels according to tangential velocity.

- In the case of *cohesionless model*, we search the concentration equilibrium. If the concentration C in a `FULL` voxel is higher than the equilibrium $\hat{C}$, we apply deposition to the underlying `MAT` voxel. If the opposite is true, we apply equation (9) to compute erosion.

3. In the last step, we solve for the material transportation within the fluid. At this point, we use the material field from the erosion step, the velocity and the pressure field. This step involves the material deposition, material transportation according to $\mathbf{u}$ and the diffusion of material within the fluid.

In the visualization phase, we use two different tools. The first tool—the OpenGL previewer—is used for interactive previews. It allows the user to detect errors and set some interesting views. We consider the materials to be semi-transparent and detect the free level using the marching cubes algorithm.[14] We can display stored data and interact with it in real time. We use a photon mapper for better display, using the regular space subdivision algorithm to accelerate the rendering. We use tricubic interpolation of the neighbouring voxels to detect free level and display the material in water by ray attenuation. We use caustic maps to display caustics. In Figure 2, we show an example of caustics rendered by the photon mapper. The first image shows the camera positioned below water, while the second image shows the camera positioned just close to the water level. Next, we'll provide some examples to illustrate the results from our algorithm.

Examples from Experiments

In this section, we describe the set of experiments performed to illustrate our algorithm. We show the simulation of erosion due to meanders, a spring of water, a waterfall, a sediment wash and a maze.

Meanders

We simulated a river bed with meanders flooded by a huge wave as shown in Figure 3. A detailed view of the simulation is presented in Figure 4.

For this simulation, we created a scene with $120 \times 32 \times 120$ voxels, with 84 source voxels. We used the cohesionless model for simulation, and assumed that the water has no initial saturation and there is only one kind of material in the scene. The water gets saturated during the erosion process.

The water fills up the existing river bed and the first small barrier is eroded due to a strong force in the perpendicular direction. The running water finds the shortest possible path from the beginning to the end, creating a new river bed and leaving the old curved parts of the river as billabongs. This phenomenon is frequently observed in nature.

Spring of Water

Figure 5 shows a spring of water in a cohesive and a cohesionless environment. The scene size is $60 \times 32 \times 60$ voxels with the size of water source 5×5 voxels. The figure shows that the cohesive material does not readily facilitate the formation of distinct streams and river beds. However, the cohesionless material, after a short simulation time, forms distinct and deep rivers to guide the water.

Waterfall

In Figure 6, we present an example of a receding waterfall. This scene is composed of $80 \times 64 \times 30$ voxels, with

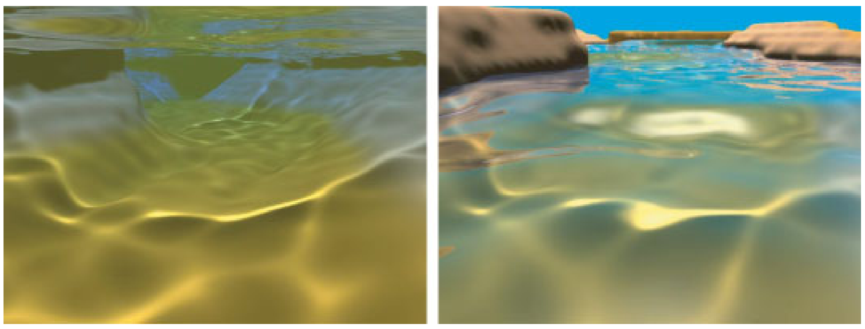


Figure 2. Rendering of caustics using a photon mapper.

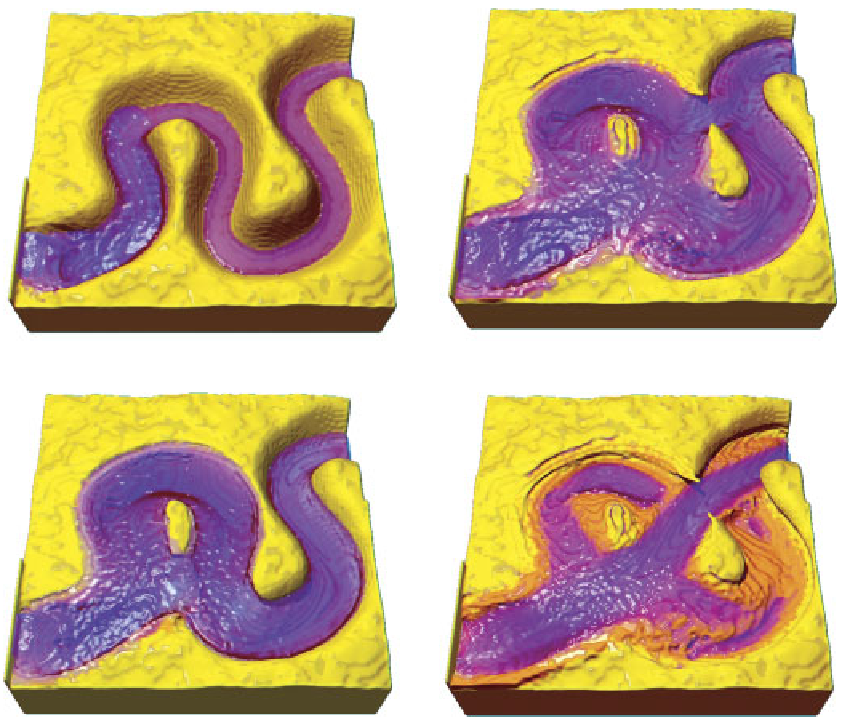


Figure 3. A huge wave erodes a meander, forming billabongs.

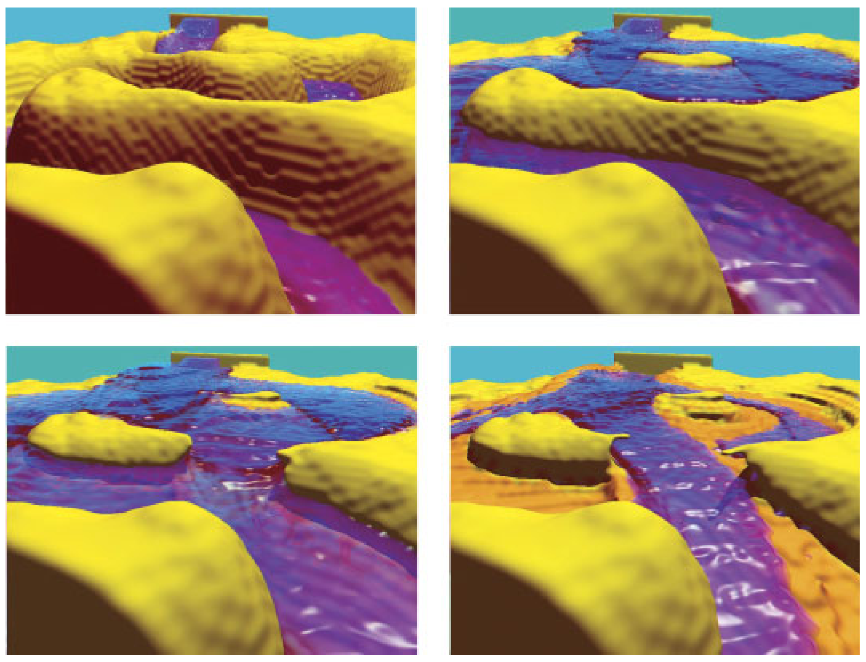


Figure 4. Running water erodes a terrain.

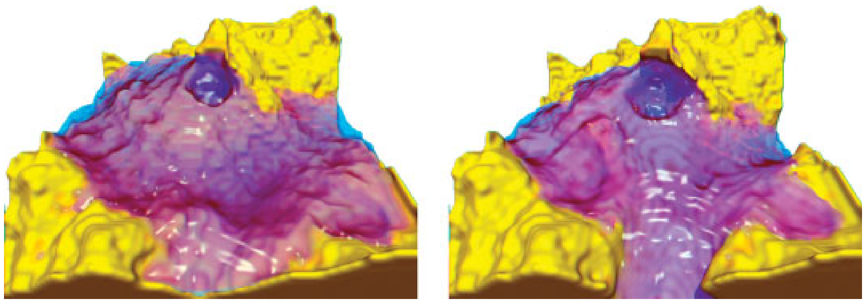


Figure 5. Spring in a cohesive (left) and a non-cohesive (right) terrain.

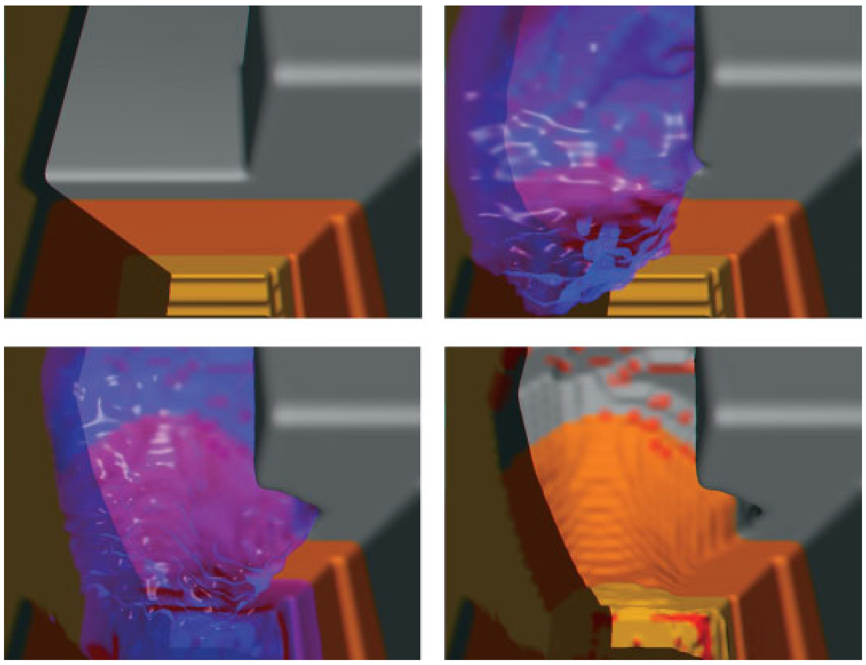


Figure 6. Receding waterfall.

50 voxels for the water source. The waterfall is shown to recede because of erosion of its base, which takes a long time in nature. The same effect can be simulated quickly by adjusting the parameters of the model.

Sediment Wash

Water running downhill forms a characteristic shape in the form of steps. For the purpose of illustrating a sediment wash, we created a scene with $120 \times 32 \times 20$ voxels. We created four source-water voxels to feed the scene with 40% saturated water.

We show the results in Figure 7. The left-hand image in Figure 7 shows the cohesive erosion of a low hill resulting in a parabola-like shape of the river bed in the direction of flow. If water flows with a high velocity, the saturated water cannot capture more material. Therefore, there is little or no erosion on the upper part of the hill, and more erosion towards the lower part where the flow velocity is low. The right-hand image in Figure 7 shows the step-like structures formed because of cohesionless erosion. This model has a strong relationship between erosion and deposition that causes step-like fluctuations around the parabolic shape of the river bed. The saturation is caused by high velocity of water flow. As a result of saturation, there is no more erosion and only deposition in some parts. This creates steps that can be observed in nature on river beds and mountain valleys. In both the cases illustrated in Figure 7, a

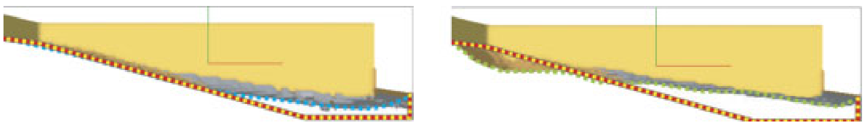


Figure 7. Cohesive and cohesionless erosion.

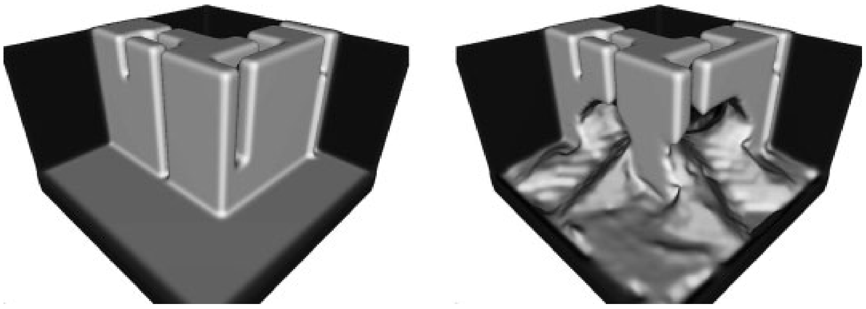


Figure 8. Erosion in a maze.

parallel cut also has a parabolical shape since the water flows slowly close to the banks and is faster in the middle.

Maze

Figure 8 displays the result of application of a strong erosion to a simple maze. Water finds its way out and erodes the paths, searching for the shortest possible route to create a balance.

The simulation times in all of the above applications range from 0.4 s to 17 s per frame using a 3 GHz PC depending on the complexity of the scene. The major factor in computation load is the number of cells that contain water and the total size of the grid.

Conclusion and Future Work

In this paper we have presented a new physically based algorithm for visual simulation of hydraulic erosion. This algorithm models the liquid motion by solving Navier–Stokes equations. It calculates the sediment dissipation and falling using material transport equation, and applies the erosion model to the boundary between water and material to simulate erosion and deposition. The application is initialized by a user-defined scene description of the water sources and drains, and by a set of parameters to define the terrain. The system is able to simulate a wide variety of scenarios, ranging from low-hill wash load to waterfalls.

We have introduced two approaches to account for cohesive and cohesionless materials. The cohesive model is good for slow water transport, and is used to simulate water step formation along the bottom of a river. Its erosion speed is relatively slow compared with the cohesionless model, which is more suitable for rapidly changing scenes such as a meander or a waterfall.

The algorithm performs well in modelling but may not be appropriate for real-time performance. We have been able to perform simulation on a 3 GHz PC, which calculated a scene with 450 k voxels at the rate of 17 s per frame. This makes it difficult to simulate vast areas, and its speed of execution constitutes our future interests.

We also note that the Navier–Stokes solver computes the solution without considering density change of the `FULL` voxels with the material.

We have used only a preliminary method for visualization as that was not our primary focus. We can use a better algorithm, for example the techniques developed by Foster and Fedkiw,[12] to display the water level in a better manner. There are also algorithms that can be used to display the material within the water with more precision.

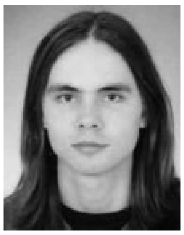
References

1. Kelley AD, Malin MC, Nielson GM. Terrain simulation using a model of stream erosion. SIGGRAPH Proceedings. *Computer Graphics* 1988; **22**(4): 263–268.
2. Musgrave FK, Kalb CE, Mace RS. The synthesis and rendering of eroded fractal terrains. SIGGRAPH Proceedings. *Computer Graphics* 1989; **23**(3): 41–50.

3. Chiba N, Muraoka K, Fujita K. An erosion model based on velocity fields for the visual simulation of mountain scenery. *Journal of Visualization and Computer Animation* 1998; **9**: 185–194.
4. Dorsey J, Edelman A, Legakis J, Jensen HW, Pedersen HK. Modeling and rendering of weathered stone. In *Proceedings of the SIGGRAPH Conference*, Los Angeles, CA, August 1999; 225–234.
5. Beneš B, Forsbach R. Visual simulation of hydraulic erosion. *Journal of WSCG* 2002; **10**(1): 79–86.
6. Li X, Moshell JM. Modeling soil: realtime dynamic models for soil slippage and manipulation. In *Proceedings of the 20th Annual Conference on Computer graphics and interactive techniques*. SIGGRAPH, ACM Press: New York, 1993; 361–368.
7. Sumner RW, O'Brien JF, Hodgins JK. Animating sand, mud, and snow. *Computer Graphics Forum* 1999; **18**(1): 17–26.
8. Onoue K, Nishita T. A method for modeling and rendering dunes with wind-ripples. In *Pacific Graphics 2000*, Hong Kong, October 2000; 427–430.
9. Onoue K, Nishita T. Virtual sandbox. In *Proceedings of IEEE 2003 Pacific Conference on Computer Graphics and Applications*, Canmore, Alberta, Canada, October 2003; 252–259.
10. Langendoen EJ. CONCEPTS—conservational channel evolution and pollutant transport system: stream corridor version 1.0. Research Report No.16, US Department of Agriculture, Agricultural Research Service, Oxford, MS, 2000. http://msa.ars.usda.gov/ms/oxford/nsl/agnps/Concepts/manual/Cover.html.
11. Foster N, Metaxas D. Realistic animation of liquids. *Graphical Models and Image Processing* 1996; **58**(5): 471–483.
12. Foster N, Fedkiw R. Practical animation of liquids. In *Proceedings of the SIGGRAPH Conference*, Los Angeles, CA, August 2001; 23–30.
13. Stam J. Stable fluids. In *SIGGRAPH 99: Proceedings of the 26th Annual Conference on Computer Graphics and Interactive Techniques*, Los Angeles, CA, August 1999; 121–128.
14. Lorensen WE, Cline HE. Marching cubes: a high resolution 3D surface construction algorithm. SIGGRAPH '87. *Computer Graphics* 1987; **21**(4): 163–169.

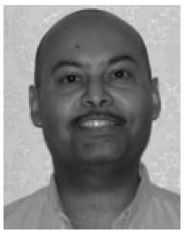
Authors' biographies:

Bedřich Beneš received his PhD in Computer Graphics from the Czech Technical University in 1998. He subsequently worked as an Assistant Professor at Czech Technical and Monterrey Institute of Technology in Mexico City. Since August 2005, he has been working as Assistant Professor in the Department of Computer Graphics Technology at the Purdue University. He is co-author of three books about computer graphics and more than 20 papers. His research interests include procedural modelling, artificial life, real-time rendering and global illumination. He is a member of ACM Siggraph, Eurographics and IEEE.



Václav Těšínský received his Master's degree in 2003 from the Czech Technical University in Prague and is presently working as a graphics software developer. He is interested in methods for procedural landscape generating, with particular emphasis in erosion processes and fluid mechanics. He has published articles on erosion simulation and volumetric data compression.

Jan Hornyš recently received his MSc in Computer Graphics at the Czech Technical University in Prague. His main areas of interests are real-time and photorealistic rendering. He is currently working as a software developer, continuing research in his areas of interest.



Sanjiv K. Bhatia received his PhD from the University of Nebraska—Lincoln in 1991. He is presently working as an Associate Professor in the Department of Mathematics and Computer Science in the University of Missouri—St Louis. His areas of research are computer graphics, image databases, digital image processing and computer vision. He has published several papers on image databases, large texture generation and its use in terrain modelling, and the application of knowledge-based techniques to information retrieval. He is also involved in sensor simulation. He is a member of ACM, Siggraph and AAAI.