Procedural generation of meandering rivers inspired by erosion

Michał Kurowski
Warsaw University of Technology, the Faculty of Electronics and Information Technology
Pl. Politechniki 1
00-661, Warsaw, Poland
M.Kurowski@ii.pw.edu.pl

ABSTRACT

This paper describes a method of procedural generation of meandering rivers inspired by erosion, which can enhance visual realism of virtual terrains. Terrain is represented using an adaptively subdivided triangle mesh with additional information (e.g. amount of soft deposit) stored in vertices. Water is simulated using Smoothed Particle Hydrodynamics (SPH), modified in order to model erosion occurring within meanders. Most experiments were performed on an initially flat terrain, so in order to provide the initial disruption of an otherwise straight flow, a simple force simulating an exaggerated Coriolis effect was introduced.

Keywords
Computer graphics, procedural terrain generation, meanders, Smoothed Particle Hydrodynamics, erosion.

1. INTRODUCTION

Creation of "virtual worlds" used in computer games, simulations, movies or art requires a significant amount of various content, including, but not limited to, textures, object models, sounds and terrains. This content can be either handcrafted by skilled professionals or generated procedurally. This distinction is not very sharp, as various packages offer the ability to combine both approaches (e.g. map editor in "Earth 2150" game), by allowing their users to procedurally generate some elements of the content (e.g. fractal terrain for use in a Real-Time Strategy game) and then manually combine and tweak them to match specific requirements (in case of the previous example: create a level patch of terrain for player's base or place resources and bridges). The growing computational power of devices in the hands of end-users is followed by their growing expectations for visual quality, which is often linked to visual complexity [Mus02]. This in turn leads to the growing amount of required work, which translates to development time, staff size and budget size, making the procedural approaches more noteworthy. Automatic generation is already very attractive to indie game developers who are aiming for low cost and low development time instead of high complexity within an acceptable budget.

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. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

Terrain is present in many "virtual worlds" and often strongly influences the final product: beautiful landscapes create atmosphere in movies and complement the action, while maps for strategy games decide the gameplay style. A complete and practical solution should be able to produce a visually complex scene in an acceptable time. This scene should contain features desired by an artist or a level designer. Unfortunately, pure procedural approaches are often either hard to control or produce results which look artificially. Methods based on physical simulations are often more intuitive and easy to integrate with other solutions, but they are also either very slow or too simplified to produce certain phenomena, such as for example meanders.

This paper describes a method that can be used to enhance an existing terrain model (created procedurally or manually) with meandering rivers, thus introducing some amount of physically inspired realism. The presented heuristic solution produces meanders by eroding the outer river banks horizontally and depositing the eroded material mainly near the inner banks. It uses SPH for water simulation and an adaptive triangle mesh for terrain representation. The most important contributions of this work are: the introduction of separate material particles, simulating an exaggerated Coriolis effect to initiate the meandering and using a low number of particles to represent water in order to achieve acceptable performance. The resulting method can be used interactively by an artist or a level designer, can take into account an existing terrain topography and can be further enhanced to allow for different material properties.

2. RELATED WORK

Due to the practical applications of procedural terrain generation many different methods were proposed. In [Mus02] Musgrave covers terrain synthesis using multifractals. Various erosion based approaches were presented in [Nei05], [Sta08] and [Kel88]. [Ben06a] by Benes is a short paper focused on hydraulic erosion introducing some problems that need to be addressed: scene size, simulation speed, user input, interactivity and covering of different scales. Some of these issues are addressed by a GPU based method introduced in [Val11], which uses an adaptively tiled "virtual layered terrain" together with pipe model for water simulation and allows interactive editing. The authors of [Cen09] present automatic terrain generation based directly on user's sketches, while Zhou, Sun and Turk propose a method which combines a "feature map" sketched by the user with example data from a digital elevation model [Zho07]. A similar idea, where a user specifies terrain primitives which are then matched against a database of "terrain units" manually extracted from real-world elevation data is presented in [Chi05]. Brosz [Bro06] proposes a method of enhancing user created model with high-resolution details extracted from another terrain. A fairly recent general overview of procedural techniques for creating 3D environments can be found in "A survey of Procedural Methods for Terrain Modelling" [Sme09].

The problem of meandering rivers is rarely addressed. One of the exceptions is work by Prusinkiewicz and Hammel [Pru93], where a river modeled using a squig curve is incorporated into a mountainous terrain created using midpoint-displacement algorithm. The process involves recursive subdivision of the triangles within the terrain's mesh and classification of the triangles' edges into river entry, river exit and neutral ones. The proposed solution produces complex fractal terrains, but lacks some realism: the river flows on constant altitude in an asymmetric valley. The authors also mention certain issues with tributaries which are not solved in the article.

Belhadj [Bel05] proposed an enhanced midpoint-displacement algorithm which is constrained by a pre-computed set of ridge lines and a river network. The network is created by tracing and combining the trajectories of randomly seeded "river particles". The results look promising and contain meander-like river paths, but their origin is not presented in the paper.

Teoh [Teo08] presented a different approach in WaterWorld. Rivers can be "seeded" by the user and then grown downward cell-by-cell in the direction of the lowest neighbor until they find another body of water. If a section of the river is found to be gently sloping, meanders are generated as an alternating curve defined by "wavelength", which is proportional to the flow of the river and "meander angle" set by the user.

In a more recent work [Teo09] the same author proposed another approach in which ridge lines are created by the user, while the river network is grown inward into the land mass from randomly placed river mouths. Each river is generated by adding consequent, randomly rotated segments and then fitting an alternating curve through them. In order to achieve varied terrain, different rivers have different "SegmentLength" and "MeanderCurvature" parameters. The heightmap is then created to accommodate the generated ridges and rivers.

The authors of [Ben06b] proposed a method based on the Navier-Stokes equations and a regular voxel grid. In one of their experiments they simulated a riverbed with meanders flooded by a wave. The simulation produced excellent results, with the river eroding the outer banks, breaking through the meanders and leaving two billabongs. However, the algorithm has high computational cost and is unsuitable for simulating large terrains (the experiment was conducted on a 120x32x120 grid). Also, the initial meanders were not created during the simulation.

A recent work on hydraulic erosion using CUDA for acceleration [Bez10], while not dealing with meanders, has some similarities with the method presented in this paper. Both use particles for water simulation and adaptive triangle meshes for terrain. However, the solution proposed by Bezin performs the subdivision during an off-line pre-processing and constrains the movement of vertices to the vertical axis (the authors use the term "adaptive heightfield").

3. PROCEDURAL MEANDERS

Proposed Mechanism

Flow of water within a meandering channel is a complex 3D phenomenon which can be resolved into a primary downstream and a secondary transverse components [Nal97]. Conducted experiments [Ram99] show that these components change with depth, distance from banks and along the length of the meander. An exact physically correct simulation (like [Ben06b]) on a large scale would require a substantial amount of computations. On the other hand, a simple curve fitting algorithms (like in [Teo08]) do not take into account local terrain variations. This paper proposes a simple erosion inspired heuristic algorithm, which is less complex than an exact simulations, but can also accommodate these variations and produces plausible results.

The presented method takes into account both erosion and deposition. The "erosion force" f_e is

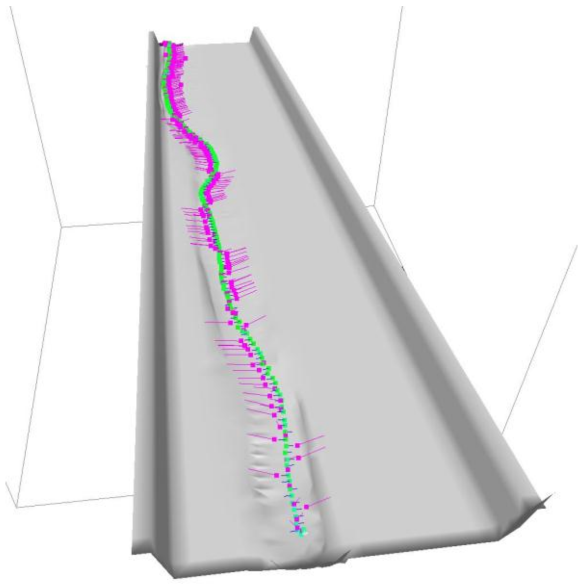

Figure 1. Attraction of the sediment to the inner banks of curves within the river channel. Green particles – water; violet particles (with vectors) – sediment.

proportional to the speed of water v multiplied by its amount w and a "directional factor", which is interpolated from 0 at the inner bank of the meander to 1 at the outer bank. The amount of material that can be potentially eroded m_p depends on the f_e, the area a for which f_e is calculated, the material's softness s and a user specified multiplier u_1:

$$m_p = f_e \, s \, u_1 \, a$$

The actual amount of eroded material m_e is m_p clamped so that water is not oversaturated with sediment. If it weren't for this condition, the multiplication and the consequent division by a wouldn't be necessary. Direction h in which eroded terrain is displaced consists of two components: downward h_y, which is constant, and horizontal $\begin{bmatrix} h_x \\ h_z \end{bmatrix}$, pointing at the outer bank. The magnitude of $\begin{bmatrix} h_x \\ h_z \end{bmatrix}$ is equal 0 where water flows straight and grows to a user defined value when the flow is curved. The terrain's displacement t is calculated as follows:

$$t = \frac{m_e}{a} \begin{bmatrix} h_x \\ h_y \\ h_z \end{bmatrix}$$

The "directional factor" ensures that the outer bank is eroded more than the inner one and $\begin{bmatrix} h_x \\ h_z \end{bmatrix}$ introduces lateral erosion, thus ensuring growth of the meander. It should be noted, that the erosion occurs only when water speed v is greater than a certain user-defined amount. This measure was introduced during early experiments in order to avoid issues with self-deepening or oscillating bed, which occurred in places where water was semi-stationary. However, further research into the mechanics of erosion revealed the existence of "critical shear stress". The introduced threshold is applied to speed, not the actual shear stress, but as both parameters are correlated, it serves the same purpose.

The emergence of meanders is the result of both the horizontal erosion of the outer bank and the deposition of sediment near the inner bank. In order to facilitate the latter process, the eroded material carried by water is attracted toward the inside of the meander (Figure 1). The strength of this attraction is user defined.

Both erosion and deposition create a positive feedback loop which causes the growth of meanders, but also requires some initial curvature. This curvature can be supplied by the shape of the terrain or by an external force. In one of his speeches, Einstein [Ein26] mentioned the role of Coriolis force in the non-uniform distribution of velocities within a flowing river. The magnitude of this force is dependent on the Earth's rotation speed, the current's velocity and the geographical latitude. In order to reduce the amount of user-controlled parameters, the presented method assumes that the changes of latitude are negligible within the simulation's domain. This leads to a simplified formula, which is intuitively consistent with Baer's law and produces satisfactory results:

$$f_c = u \begin{bmatrix} 0 \\ 0 \\ v_x \end{bmatrix} m \quad ,$$

where f_c is the force, u is a user defined factor, v_x is the velocity of water projected onto the X axis and m is the mass of water to which the force is applied.

Water Simulation

The presented method requires the water simulation to provide not only the speed and mass of water in certain areas, but also information whether the stream is curving and in what direction. The process of meandering depends on small, smooth variations in the direction of river flow, which should not be dampened in time. For performance reasons, the simulation should be conducted only in areas containing water, which usually occupy a fraction of the entire domain.

Existing water simulation methods can be divided into Eulerian (operating on meshes or fixed grids) and Lagrangian (operating on particles). Although hybrid methods exist and are used in computer graphics, erosion simulation usually employs pure approaches ([Ben06b] and [Ben09]). The Eulerian methods usually produce excellent, physically correct results, but are computationally expensive, especially when applied to large 3D domains. They also reveal a tendency to dampen movement which isn't aligned with the used grid. There are works which solve some of these problems by introducing non-regular adaptive meshes or additional particles, but they are usually complicated. Interaction with solid boundaries is either complex or requires high resolution. On the other hand, even simple Lagrangian methods do not suffer from anisotropy and can be easily optimized for large, sparse domains. Current consumer hardware is capable of simulating tens of thousands of particles at interactive rates using GPGPU [Gos10]. Physical accuracy of the simplified methods doesn't match that of the Eulerian ones, but it is usually good enough for applications in computer graphics.

The presented paper uses Smoothed Particle Hydrodynamics, which was introduced in [Mul03] and combines relative simplicity with great flexibility. The characteristic feature of this method is that certain quantities (e.g. pressure) defined at discrete particle locations can be evaluated also in their neighborhoods as continuous values. This is achieved by accumulating contributions from individual particles weighted by radial symmetrical smoothing kernels. The SPH particles used in this paper are enhanced with additional quantities for use only in the erosion algorithm – scalar "curve accumulator" c_a and vector "curve direction" c_d. c_a is defined initially as 0 for each particle. If a particle's horizontal velocity vector changes its direction more than a certain value, c_a is increased or decreased depending on whether the velocity was changed to the left or to the right, while c_d is set to left. If the absolute value of c_a is greater than a certain threshold, the stream is assumed to be curving in the direction defined by $c_a\, c_d$. This value is then used as $\begin{bmatrix} \mathrm{h_x} \\ \mathrm{h_z} \end{bmatrix}$. The performance of SPH method relies on fast finding of neighboring particles. A simple regular 2D grid containing indices of particles within a certain volume is used for this purpose.

Sedimentation

Eroded soil is carried with water as sediment. In other works using SPH this information is usually embedded in water particles. The sediment flow and concentration is influenced not only by the water velocity, but also for example by gravity and diffusion. The authors of [Ben09] solve this problem using a donor-acceptor scheme, where the movement of imaginary sediment particles is simulated by material transfer between water particles. This approach produces realistic results, but depends on a large amount of particles and allows the sediment concentration to be defined only in and between them. The method described in this paper strives for river representation using the lowest possible amount of particles and requires the sediment to be attracted towards the inner banks, so that the center of its concentration may not overlap with the center of water mass. For this reason the presented solution uses separate sediment particles that are influenced by the flow of water, gravity and attraction to the insides of the meanders.

Each particle's speed v is calculated as:

$$v_{n+1} =$$
$$v_n + (g + r + a(s - v_n) + normalize(\begin{bmatrix} \mathrm{h_x} \\ 0 \\ \mathrm{h_z} \end{bmatrix})fs)t,$$

Where v_{n+1} is the particle's speed in the next simulation step, v_n is the particle's speed in the current simulation step, g is the gravity vector, r is the strength of repulsion from the terrain, a is a user defined "acceleration factor", f is a user specified attraction factor, s is the average speed of the surrounding SPH particles and t is the time delta between the simulation steps.

If new sediment is added to the simulation, the nearest sediment particle is searched for. If such particle is not found, then a new one is created. If the search is successful, a certain amount of the sediment is added to the existing particle, so that the capacity of the particle is not exceeded. The amount that could not be added is returned to the erosion algorithm and is used to clamp the m_p value introduced earlier. This stops erosion when water is oversaturated with eroded material. However, the oversaturation can occur if multiple particles concentrate in a small volume (there are no collisions between them, so they can be arbitrarily close to each other). In this case the sedimentation process is initiated to get rid of the excess material. The sedimentation also occurs as the particle ages, when it loses contact with water (which rarely happens) or its speed is lower than a threshold v_t, calculated based on user-controlled parameter u_2 and the distance from the center of water mass c_w:

$$v_t = u_2\left(1 + \frac{4}{r}\|p - c_w\|\right),$$

where p is the particle's position and r is the smoothing radius of the SPH simulation. This formula is purely heuristic and was developed by experimentation. The situation in which the sediment particle leaves water is undesired, so it's movement is restricted to the volume within $0.99r$ of the nearest SPH particle. This solution works most of the time, but breaks if water velocity changes rapidly. However, these rare stray sediment particles do not seem to introduce any problems to the simulation. For performance reasons, if two particles are close enough and do not exceed the saturation limit, they are merged together into a larger one. The search for neighbors in a given area is accelerated by a 2D grid identical to the one used in water simulation.

Terrain Representation

The author's first experiments with meandering were conducted on a commonly used heightmap (Figure 3). The directional feature of the erosion was simulated by eroding two points at the same time – the one where the erosion parameters were calculated and its neighbor in the direction of the erosion. The meanders started to form, but their evolution ended fast. When the elevation difference between the river bed and its bank increased, the simplified lateral erosion was less effective, while the sedimentation performance was constant. This lead to silting and overflowing of the bed. A better terrain representation was required.

Further experiments were conducted using "deformable voxels" introduced by the author in [Kur11a]. The initial results were encouraging, but as it turned out, it was difficult to maintain coherence of the grid. An attempt to refine the terrain representation was made. However, before a satisfactory solution was found, rising computational complexity made the idea impractical and without much room for optimization.

Because of the difficulties with voxel representations, a polygonal mesh was used. The collision surface is defined by triangles, while the properties of the terrain are stored within vertices, which are displaced by erosion and sedimentation. Triangles are adaptively subdivided (Figure 2) in order to ensure, that no water particle can touch a triangle without influencing the erosion of at least one of its vertices. One of the obvious advantages of such a mesh is that any surface can be easily represented. There are however some important issues to deal with. If some vertices are moved further apart, then some others are moved closer together. This results in an oversampled mesh and while it doesn't pose any problems to the correctness of the simulation, it can significantly reduce the performance. Smooth merging of triangles without causing sudden changes in terrain geometry is yet to be implemented in this solution. The mesh can become degenerated when two surfaces get close enough to penetrate each other. This issue was solved in [Mul09], but can be difficult in case of an almost unconstrained mesh with arbitrary resolution used in the presented paper. A simple solution requires the previous problem to be addressed first. However, this degeneration seems to occur only in the last stages of a meander's evolution, so while it certainly limits the simulation, it is still possible to achieve decent results. Additional issues occur if the vertices defining consequent triangles on the river bed are at

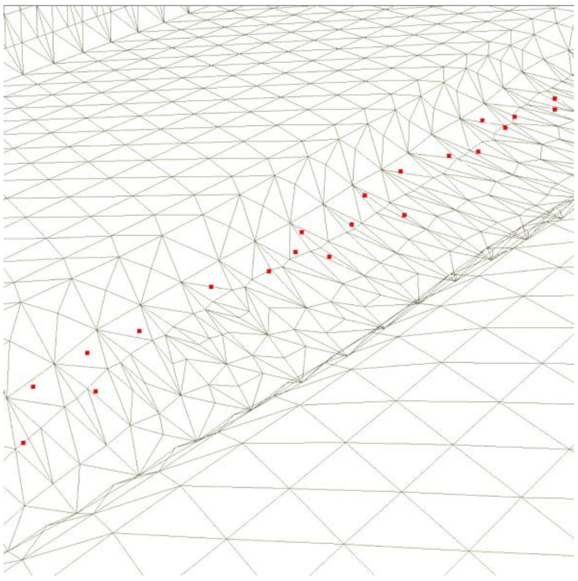
Figure 2. Terrain represented using a triangle mesh. The river channel is adaptively subdivided.

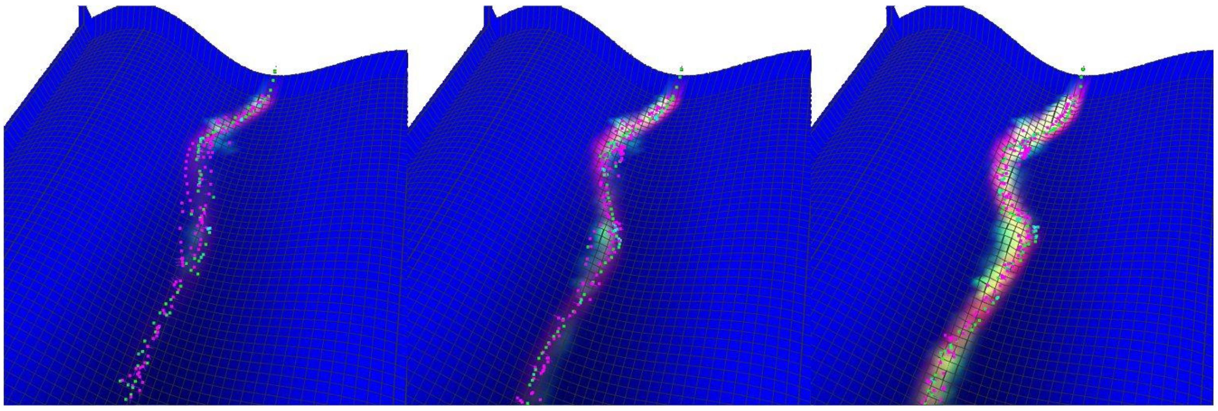
Figure 3. Evolution of a meander on a heightmap terrain (green particles – water; violet particles – sediment; green terrain – deposition; red terrain - erosion). The initial bends are due to the simulated Coriolis force and shape of the terrain. The positive feedback loop ensures the growth of the meander, which is unfortunately slower and slower, until it stops completely. It should also be noted, that the bend closest to the source is the most prominent, which is undesirable.

significantly different distances from the flow of water. This results in irregular erosion, which produces uneven bed surface which significantly slows the flow of water, creating unwanted small ponds. One solution is to introduce denser subdivision, which results in poorer performance. Other option which was also used was to perform smoothening of the eroded terrain after each 10 simulation steps.

The amount of eroded or deposited material is proportional to the magnitude of the displacement and the area supporting the displaced vertex. The deposition is always upwards and the erosion is assumed to be mainly downwards. Therefore the area is calculated as a sum of areas of all the triangles to which the given vertex belongs, projected on the horizontal plane.

Usually a hard bedrock or cohesive soil is harder to erode than fresh sediment. In order to take into account this feature, each vertex stores the thickness of a "soft layer". It indicates what part of the vertex displacement is due to the accumulation of material that was eroded elsewhere. If this value is larger than 0, we assume that we are eroding soft, fresh sediment. The introduced earlier material softness s of the base terrain is set to 1, while s of this layer is controlled by the user. While the thickness of the soft layer is increased by deposition, it is decreased both by erosion and time. The latter is introduced to simulate the hardening of the sediment into rock. The soft layer prevents sedimentation in places which are being constantly eroded and facilitates the transport of the material.

In order to facilitate the performance of adaptive subdivision, area calculation and collision detection, vertices are indexed in a 3D grid and contain information about all their neighbors and all the triangles they belong to.

4. SIMULATION SOFTWARE

The simulation software was written in C++ and uses FLTK library for user interface, OpenGL for visualization and OpenMP for parallel computations, so that it can take advantage of the modern multi-core processors. These libraries were chosen to enable the application to be compiled and run both on Windows and Linux systems. Water simulation, sediment transport and terrain representation were carefully separated in order to enable easy swapping of different terrain implementations. The code uses lambda functions introduced in the new C++11 standard. While they do not have any noticeable impact on performance, they greatly reduced the time needed to implement and test various approaches without compromising the introduced separation.

Water sources and sinks can be added, modified or removed before and during the simulation. SPH, erosion and deposition parameters can also be adjusted in run-time.

5. RESULTS

First experiments were conducted in an artificial valley with cross-section in the shape of a flattened sinusoid. Meandering in such a terrain occurs naturally if the stream of water is not perfectly aligned with the valley. However, many rivers meander on flatlands and so further experiments were conducted on a completely flat surface (4096 x 768 units in size) surrounded by barriers from 3 sides. In order to enforce the flow of water, the surface was tilted towards the side without any barrier. An artificial source producing one particle per second at a random position within a radius of 10 units was placed on the top. Water and sediment that fell from the surface after reaching its lower end was removed by an artificial sink. The created system was thus open. The CM factor regulating the force induced by Coriolis effect was set to 0.005. SPH radius was 25 units. Other user-controlled variables (over 20 in total, their complete description is beyond the scope of this article) were different in consequent experiments and sometimes tweaked in run-time.

The simulation contained usually around 300 SPH particles and a similar - sometimes slightly larger - amount of sediment particles. One step of the simulation took around 0.02s on Intel Core 2 Quad 2.66 GHz with 4GB of RAM. The approximate time spent in different subsystems is as follows: drawing ~ 1%, erosion ~ 25%, adaptive subdivision of terrain ~ 5%, SPH simulation ~ 30% (~25% is spent in calculating collisions with ground), sediment transport and deposition ~30%. Precise percentages depend strongly on the state of the simulation.

The artificial river bends, the outer banks are intensively eroded and the deposition occurs mainly on the inner ones. Meanders start to form and then grow (Figure 4). Dense subdivision is visible within the river channel, especially on the outer banks. The horizontal shape of the river looks convincing. However, the cross-section reveals one unwanted feature – the smoothing of the eroded terrain introduced to ensure the undisturbed flow of water causes the river channel to be too flat, lowering the bed near the inner banks (where most deposition occurs) and raising the bed near the outer ones (Figure 5).

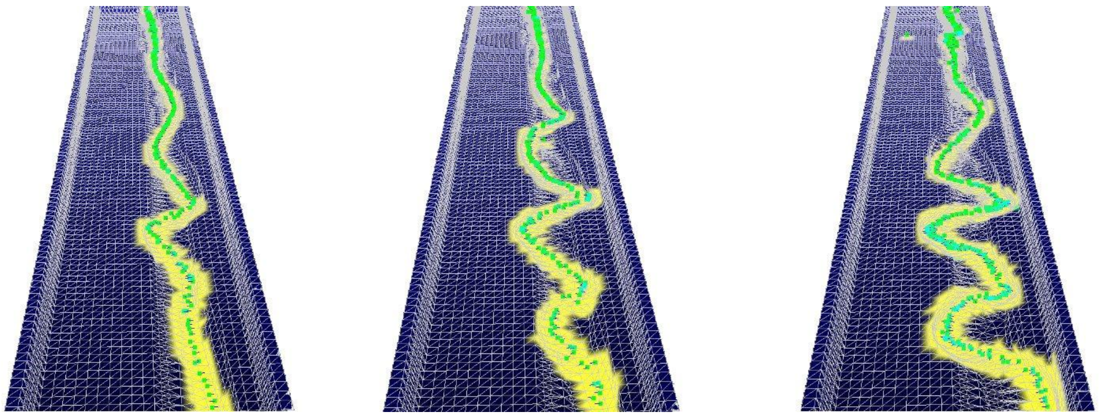

Figure 4. Meanders generated using the proposed method using a triangle mesh terrain representation. Green particles denote water and the yellow area corresponds to the river channel. It should be noted that the curves started to form due to the introduced Coriolis effect on an initially flat surface. The further downstream, the larger the meanders are, which gives them a plausible appearance.

6. CONCLUSIONS

The main goal of the experiments – to produce growing meanders – was achieved. Introduction of the simplified Coriolis force lead to satisfactory results. Triangle mesh seems to be the best choice for this type of simulation, but there are certain issues that require more work. In order to ensure the proper shape of the cross-section, the smoothing algorithm needs to be refined or completely replaced by a better solution with a possibly low negative effect on the performance. An effective solution to the mesh degeneration must be implemented in order to enable the full evolution of meanders up to the formation of billabongs. Additional tests on more complicated terrains should be conducted in order to find and resolve possible issues.

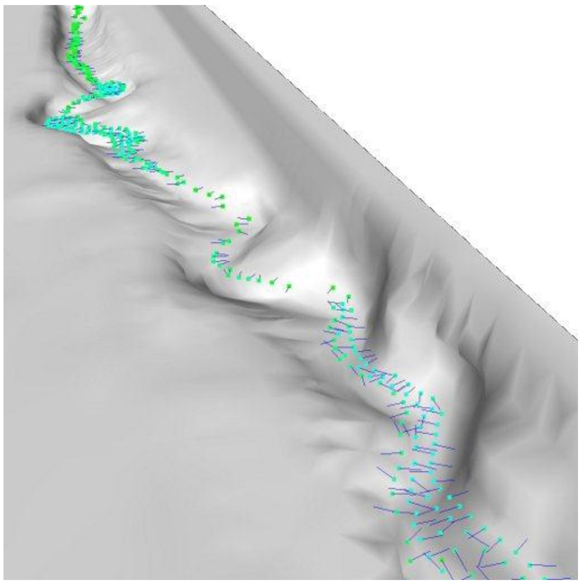

Figure 5. Close-up of the river channel (terrain surface is rendered using Gouraud shading with a single light source at the camera's position). The bed is too wide and too flat.

After refining the existing functionality, certain features will be added. At this moment the only sources of water are artificial and placed by the user. This results in the amount of water being approximately constant along the channel's length, while natural rivers tend to be small at the source and then grow larger due to additional supply from tributaries. Such tributaries should be automatically generated based on a rainfall simulation, probably similar to [Teo08]. The present method assumes the same material softness for the entire base terrain. It was shown in [Kur11b] and [Ben01] that introduction of several, possibly intersecting, layers of terrain with different parameters may significantly enhance the results of erosion. A similar feature should be implemented using the triangle mesh.

The SPH simulation was planned to be migrated to a GPGPU solution like OpenCL or CUDA. However, the solver uses just a few percent points of the processing time, so the expected benefits would be negligible. Possible performance related optimizations (and potential porting to GPGPU) should concentrate on the terrain representation and collision detection instead.

The proposed method has also an interesting feature compared to purely random algorithms – the emergence and growth of the meanders is a continuous process, which could be presented to the end-users as a feature. "From Dust" is a game in which the player assumes the role of a god and achieves the mission objectives by shaping the landscape using nature elements. One of these elements is water, which erodes the terrain. Adaptation of the proposed method for a direct use in a game environment would require a significant amount of work, but should be possible and could enhance the gameplay.

7. REFERENCES

[Bel05] Belhadj F. and Audibert P.. Modeling Landscapes with Ridges and Rivers. Proceedings of the ACM Symposium on Virtual Reality Software and Technology 2005, 2005.

[Ben01] Benes B. and Forsbach R.. Layered Data Representation for Visual Simulation of Terrain Erosion. IEEE SCCG2001 Budmerice, Slovakia, 2001.

[Ben06a] Benes B.. Hydraulic Erosion: A Survey. Invited paper to SCCG 2006, ACM SIGGRAPH, 2006.

[Ben06b] Benes B., Tešinsky V., Hornys J. and Bhatia S.K.. Hydraulic Erosion. Computer Animation and Virtual Worlds 17(2), 2006.

[Ben09] Kristof P., Benes B., Krivanek J. and Stava O.. Hydraulic Erosion Using Smoothed Particle Hydrodynamics. Proceedings of Eurographics 2009 vol. 28 No.2, 2009.

[Bez10] Bezin R., Peyrat A., Crespin B., Terraz O., Skapin X. and Meseure P.. Interactive hydraulic erosion using CUDA. Proceedings of the 2010 international conference on Computer vision and graphics: Part I, 2010.

[Bro06] Brosz, J., Samavati, F. and Sousa, M.. Terrain synthesis by-example. Advances in Computer Graphics and Computer Vision International Conferences VISAPP and GRAPP 2006, 2006.

[Cen09] Puig-Centelles A., Varley P.A.C. and Ripolles O.. Automatic Terrain Generation with a Sketching Interface. Proceedings of the 17th International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision (WSCG ’09), 2009.

[Chi05] Chiang, M.Y., Huang, J.Y., Tai, W.K., Liu, C.D. and Chiang, C.C.. Terrain synthesis: An interactive approach. Proceedings of the International Workshop on Advanced Image Technology, 2005.

[Ein26] Einstein, A.. The cause of the formation of meanders in the courses of rivers and of the so-called Baer's Law. Read before the Prussian Academy, January 7, 1926, published in Die Naturwissenschaften, Vol. 14 (English translation in "Ideas and Opinions," by Albert Einstein, Modern Library, 1994), 1926.

[Gos10] Goswami P., Schlegel P., Solenthaler B. and Pajarola R.. Interactive SPH Simulation and Rendering on the GPU. Proceedings ACM SIGGRAPH/Eurographics Symposium on Computer Animation, 2010.

[Kel88] Kelley A.D., Malin M.C. and Nielson G.M.. Terrain simulation using a model of stream erosion. Proceedings of SIGGRAPH '88, 1988.

[Kur11a] Kurowski M.. Modelowanie terenu na bazie symulacji erozji z wykorzystaniem deformowalnych wokseli. Zeszyty Naukowe Wydzialu Elektroniki, Telekomunikacji i Informatyki Politechniki Gdanskiej, Proceedings of WGK 2011 vol. 1, 2011.

[Kur11b] Kurowski M.. Modelowanie terenu 3D z jaskiniami inspirowane erozją. Zeszyty Naukowe Wydzialu Elektroniki, Telekomunikacji i Informatyki Politechniki Gdanskiej, Proceedings of ICT Young 2011 vol. 1, 2011.

[Mul03] Müller M., Charypar D. and Gross M.. Particle-based fluid simulation for interactive applications. SCA '03 Proceedings of the 2003 ACM SIGGRAPH/Eurographics symposium on Computer animation, 2003.

[Mul09] Müller M.. Fast and Robust Tracking of Fluid Surfaces. Proceedings of ACM SIGGRAPH / EUROGRAPHICS Symposium on Computer Animation (SCA), 2009.

[Mus02] Ebert D.S., Musgrave F.K., Peachey D., Perlin K. and Worley S.. Texturing and Modeling, Third Edition: A Procedural Approach. The Morgan Kaufmann Series in Computer Graphics, 2002.

[Nal97] Nalder G.. Aspects of Flow in Meandering Channels. Transactions of the Institution of Professional Engineers New Zealand: General Section Volume 24 Issue 1, 1997.

[Nei05] Neidhold B., Wacker M. and Deussen O.. Interactive physically based Fluid and Erosion Simulation. Proceedings of the Eurographics Workshop on Natural Phenomena, NPH 2005, 2005.

[Pru93] Prusinkiewicz P. and Hammel M..A Fractal Model of Mountains with Rivers. Proceedings of Graphics Interface’93, 1993.

[Ram99] Rameshwaran P., Spooner J., Shiono K., and Chandle, J.H.. Flow Mechanisms in two-stage meandering channel with mobile bed. Proceedings of IAHR Congress in Graz, Austria, 1999.

[Sme09] Smelik R.M., Kraker K.J., Groenewegen S.A., Tutenel T. and Bidarra R.. A survey of Procedural Methods for Terrain Modelling., CASA Workshop on 3AMIGAS, 2009.

[Sta08] Stava O., Benes B., Brisbinn M. and Krivanek J.. Interactive Terrain Modeling Using Hydraulic Erosion. Eurographics/SIGGRAPH Symposium on Computer Animation, 2008.

[Teo08] Teoh T.S.. River and Coastal Action in Automatic Terrain Generation. Proceedings of the International Conference on Computer Graphics and Virtual Reality’08, 2008.

[Teo09] Teoh T.S.. RiverLand: An Efficient Procedural Modeling System for Creating Realistic-Looking Terrains. ISVC ’09 Proceedings of the 5th International Symposium on Advances in Visual Computing, 2009.

[Zho07] Zhou, H., Sun, J., Turk, G. and Rehg, J.. Terrain synthesis from digital elevation models. IEEE Transactions on Visualization and Computer Graphics 13,2007.

[Van11] Vanek J., Benes B., Herout A. and Stava O.. Large-Scale Physics-Based Terrain Editing Using Adaptive Tiles on the GPU. IEEE Computer Graphics and Applications November/December 2011, Vol 31, No 6, 2011.