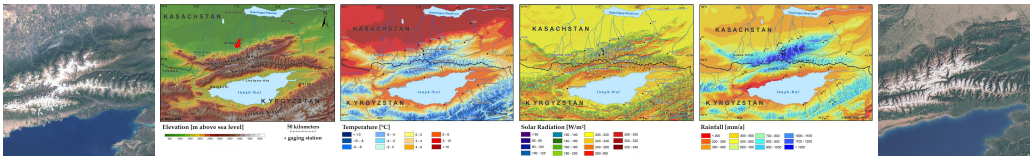
Procedural Reproduction of Terrain Textures with Geographic Data

Carsten Dachsbacher, Tobias Bolch, Marc Stamminger

Procedural Reproduction of Terrain Textures with Geographic Data

Carsten Dachsbacher[1,3], Tobias Bolch[2], Marc Stamminger[3]
[1] REVES INRIA Sophia-Antipolis
[2] Institute for Cartography, Technische Universitaet Dresden
[3] Computer Graphics Department, University of Erlangen-Nuremberg



Abstract

Surface textures of high resolution and quality, either acquired from aerial or satellite imagery or computed using procedural models, are crucial for photorealistic terrain rendering. Procedural models provide a compact representation and can be evaluated at run-time. In this paper we present an extension to an existing, GPU-friendly procedural texturing model, such that it can be fitted semi-automatically to real-world data. In order to increase realism and to account for geographic conditions, we also include temperature, solar radiation and rainfall distributions – simulated or modeled using measured data from gaging stations – into the reproduction process. The original surface texture is no longer required for rendering: instead a new texture of arbitrary resolution is synthesized at runtime. In addition to the compact procedural model we store elevation data, anyway required for the terrain rendering, and low-resolution geographic data. We show results of our method applied to a comparatively little cultivated region in Central Asia.

1 Introduction

Surface textures of high resolution and quality are crucial for photorealistic terrain rendering. Basically they can originate from two sources: either they are acquired as real-world data, e.g. from satellite imagery, or they are created from the input elevation data using appropriate procedural models. In both cases the memory consumption of the acquired or precomputed textures can be tremendous if great viewing distances or high surface detail for close views is provided and therefore adequate implementation strategies have to be applied to achieve interactive rendering.

Procedural models for terrain texturing can be used to create completely artificial, yet realistic landscapes. In this paper we propose means to mimic real terrain textures (left image in the teaser) by guiding a procedural model with real-world data. We incorporate further geographic input data in addition to elevation data, namely rainfall, solar radiation, and temperature (center images in the teaser). The terrain surfaces visible in a georegistered satellite image are classified and the geographic preconditions for their appearance are estimated. This information can be used to reproduce the input texture with the advantage of a compact procedural description (right image in the teaser). Using a model that can be evaluated at arbitrary resolutions, we can also render close-ups of the terrain by adding procedural detail.

This paper is structured as follows: in Section 2 we give an overview over related and previous work and Section 3 gives an introduction to geographic considerations for realistic terrain texturing. Section 4 recapitulates the basics of the procedural model which serves as a starting point for this work. Section 5 describes our texture reproduction by guidance of the procedural model followed by results and conclusions in Section 6.

2 Previous Work

Diverse caching and compression strategies have been developed to deal with huge amounts of precomputed or acquired texture data. Two approaches with hardware support are *clipmaps* [SGI] (on SGI's InfiniteReality) and *S3 texture compression* (see e.g. [Mic02]) supported by all contemporary graphics hardware. Other solutions to this problem are solved on the software side using adequate caching or paging strategies: typically a large terrain is rep-

resented by a quadtree-like data structure and the required texture (resolution) for each visible node is determined and the appropriate texture is loaded or generated. Examples for this technique have been presented among others by Blow [Blo98, Blo00b] and Ulrich [Ulr02].

However, in various settings, it is not possible to use acquired real-world data for terrain rendering. It may be that the texture resolution is too low for close-up views or that the images include human housing or agriculture although undisturbed, natural scenes are to be rendered. Apart from that, it is often the case that storage or transfer of the textures is not desirable – then, procedural models can be used to compute textures.

Many texturing models, mainly those often used in computer games and other interactive applications, are computationally rather cheap but lack realistic appearance. For example procedurally precomputed low-resolution texture maps combined with detail or bump maps are not sufficient for a plausible appearance of soil, especially due to the visible repetitions. Although Wang tiles [CSHD03] can solve this problem they require a set of precomputed texture tiles and cannot account for varying terrain features, like steep slopes and plains. Precomputing procedural textures at high resolution requires paging and caching algorithms, but even those have their limitations. Widely used approaches in computer games are *texture splatting*, that is compositing of a terrain texture by blending a set of tileable textures (e.g. for rock, sand and grass surfaces) in the frame-buffer (see Bloom [Blo00a] for details) or *texture bombing* (see [Gla04]). A procedural model which can be computed efficiently at arbitrary resolutions using programmable graphics hardware and which accounts for terrain features was introduced by Dachsbacher et al. [DS05]. It is comparable to the model used in Terragen [Fai05] in terms of visual quality and flexibility, but can be evaluated at run-time at interactive frame rates.

For reproducing textures, the input data has to be interpreted. Premoze et al. [PTS99] propose methods for classifying surface types and for removing shading and shadowing artifacts from aerial imagery. Using this information, new views of the terrain for arbitrary times of day and atmospheric conditions can be rendered.

Much research has been spent on terrain rendering, level-of-detail algorithms and texturing. As these topics are not closely related to this work, we direct the interested reader to `http://www.vterrain.org` which is a valuable source of links and information.

3 Geographic Considerations

Apart from efficient computation, the quality of a procedural model depends mainly on the realism it achieves. Those procedural models taking terrain features into account so far only consider terrain elevation and slope (e.g. the Terragen [Fai05] model). We propose to include additional, geographic input data, e.g. temperature, solar radiation and rainfall, into the procedural model. This enables us to steer a procedural model such that it reproduces a given input texture. This is not possible when regarding only input elevation data, as terrain surfaces and particularly vegetation also depends on other factors. The geographic quantities can either be acquired from appropriate gaging stations or computed by simulations.

In this work we concentrate on generating textures only, which implies a certain viewing distance. For close views, vegetation and small scale detail, e.g. rocks, have to be represented geometrically. Deussen et al. [DCSD02] and Colditz et al. [CCDH05] propose methods to handle the complexity of such ecosystems for rendering. To account for additional input data for texturing, it is important to know how it influences the appearance of surface types. We refer the interested reader to physical geography and biogeography textbooks (e.g. [MH05, Pid05]) and present a collected summary of aspects that proved to be decisive.

Erosion simulation can be used to determine where solid rock and where loose sediments exist. However, erosion itself is a quite complicated process, if vegetation and non-uniform material is considered. The basis of erosion and rockfall is weathering or decomposition, e.g. due to frost weathering, chemical decomposition or plant roots. Rockfalls are common for high mountains and exhibit brighter tear-off edges, due to non-weathered stone. Besides from terrain slopes, soil properties and rainfall, erosion also depends on vegetation, as plants prevent from soil loss. Vice versa, the plant distribution also depends on surface materials and thus on the result of erosion.

The appearance of plants in general depends on few parameters: they require sufficient water sup-

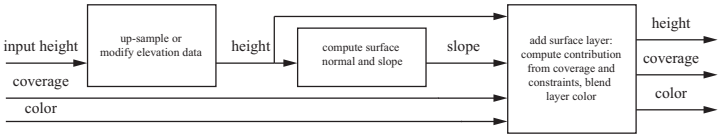


Figure 1: Applying a surface layer includes modification of the elevation data, normal/slope computations, and finally the determination of the new surface color and height value.

ply, an adequate soil to strike roots and a certain amount of solar radiation for photosynthesis. On the other hand the solar radiation is an important steering factor for the evaporation; thus it is also a factor that limits the growing of plants. The sunshine duration can be computed for a given terrain and location on the earth, provided that an average cloud cover can be assumed. Computing the irradiation for a surface location from the *solar constant* (the amount of energy received at the top of the earth's atmosphere, $1368W/m^2$) is explained by Stickler [Sti99]. Concerning the availability of water, it is important to note that it does not depend solely on direct rainfall, but also on subterranean streams that are hard to simulate. The rainfall distribution depends mainly on large-scale terrain features. The precipitation normally rises with the elevation due to air cooling and therefore high mountains receive more rainfall than surrounding planes. In addition, an important aspect is the rainfall 'shadowing' from mountains in the main wind direction. The occurrence of perennial snow and ice also depends on certain climatic conditions.

Procedural models for height field generation and real world elevation data usually do not exhibit soil type information. For this, the distribution of soil materials is usually computed within the procedural texturing model assuming that e.g. rocky surfaces appear at steep slopes. The Cached Procedural Texturing (CPT) model [DS05] is constructed such that one surface type is the precondition for the existence of another: for example, a surface type representing fertile sediments is the precondition for grass and other plants to appear. So far, however, only altitude and slope criteria have been considered for the distribution of soil materials.

This paper describes how to extend and fit the CPT model semi-automatically to real-world data, e.g. acquired satellite images. By this, the original surface texture is not required for the rendering and is replaced by a very compact procedural approximation which can be evaluated directly at run-time using stored low-resolution geographic data.

4 Cached Procedural Textures

As our reproduction of natural terrain textures is an extension to the CTP method, we recapitulate the basic ideas in this section. This technique generates surface textures from input elevation data while the surface appearance is described by hierarchically ordered surface layers. A surface layer represents a soil or vegetation layer and can contribute to the final texture color at a certain location when constraints for appearance (e.g. a certain range of elevation) depending on user-defined layer parameters and input elevation data are met. Originally, this method was designed to compute artificial terrain textures with few, intuitive parameters.

The method can be best explained, when assuming that we compute a texture for the whole terrain with a 1:1 texture-to-heightmap texel ratio. At first, the terrain texture is initialized with a base color representing the underlying rock material. Then one surface layer at a time is processed: for each texel of the terrain height map the layer's constraints (see Section 4.1 and 4.2) are evaluated. The surface layer only contributes to the terrain color and may change the terrain surface (modifying its elevation data), if its constraints are met. If one of the constraints is unmet, the layer is discarded at this texel. The next surface layer operates on the output color and elevation data of the previous step. After applying all layers, the final height is used to compute the terrain's surface normals for lighting computations and for displacement mapping. This technique thus produces modified elevation data through displacements, a corresponding normal map, and a color texture that represents the final terrain color.

In order to provide level-of-detail rendering a quadtree data structure is used for spatial subdivision of the terrain domain. For geometric level-of-detail a Chunk-LOD approach [Ulr02] is used. As terrain closer to the viewer requires higher texture resolutions than terrain further away, the algorithm operates on texture tiles (small textures associated a quadtree node) stored in a texture atlas [Wlo05].

The generated surface textures are cached in the atlas to exploit frame-to-frame coherency. Tiles may become invalid as the current view changes or as the current view requires them to be of different resolution. With caching, only a few or even no texture updates are required for rendering a new frame.

4.1 Surface Layers and Attributes

Whether a surface layer contributes to a surface location or texel $\mathbf{x}$ of the terrain texture depends on the evaluation of its constraints. A layer appears only in a certain range of height and slope values and the amount of its contribution to the texture color is called *coverage* $C(\mathbf{x}) \in [0;1]$. For efficient evaluation these coverage functions are computed from simple, clamped hat functions (provide continuous transitions) $h_A(x)$ and $h_S(x)$ for height and slope constraints (see Fig. 2). Please note, that h_A and h_S are functions of scalar input and output – however, we will apply them to input data retrieved from the elevation data $H(\mathbf{x})$ and define $h'_A(\mathbf{x}) = h_A(H(\mathbf{x}))$ and analogous for h'_S.

In addition to these hat functions the coverage value is slightly modified by a noise function to diversify the terrain texture which may be of much higher resolution than the input height field. Each surface layer stores parameters for evaluating the hat and noise functions, and a RGB color value representing the surface material or vegetation (see Table 1). These parameters are chosen by the user and allow an intuitive modeling of surface texturing.

4.2 Evaluation

The hierarchical structure of surface layers, also defined by the user, specifies the order of evaluation of the surface layers (Fig. 4). Layers are evaluated in depth-first order, and layers whose parents fail to satisfy their constraints cannot contribute. A layer cannot contribute more than its parent layer and to achieve this, the coverage $C(\mathbf{x})$ of a layer is bounded by its parent's coverage $C_P(\mathbf{x})$. Please note, that coverage is only propagated to a layer's children, the modified elevation data is always taken as input for the next layer (Fig. 4).

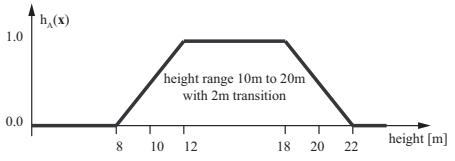


Figure 2: Surface layer constraints are represented by clamped hat functions.

The root node is special in that it has no constraints: the base material of the terrain is present everywhere. The input to the root node is the original terrain elevation data and as the target resolution of the terrain texture is usually higher, the elevation data is upsampled by cubic B-spline interpolation. Simpler bilinear interpolation leads to piecewise linear patches and causes undesirable texturing and shading artifacts.

The constraint hat functions can be efficiently computed by the clamped minimum of two linear functions. As a consequence the native instructions of a GPU's pixel shader allow a very simple implementation and a parallelized evaluation for multiple constraints. In order to provide the option of making the surface layer distribution more diversified, the noise function $N(\mathbf{x})$ with a codomain of $[-1;1]$ is scaled, biased and used to modify constraint results. The coverage $C(\mathbf{x})$ is then computed from all constraint functions (denoted by F_i here, $F_0 = h'_A$, $F_1 = h'_S$ and n being the number of constraints) and the parent's coverage:

$$C(\mathbf{x}) = clamp\left\{[b_s + s_s N(\mathbf{x})]\prod_{i=0}^{n-1} F_i(\mathbf{x})\right\} C_P(\mathbf{x})$$
$$\text{with } clamp(x) = \max\{0, \min\{1, x\}\} \tag{1}$$

Finally, the new color due to application of this layer and the new height value $H'(\mathbf{x})$, modified by a displacement map $D_S(\mathbf{x})$ (representing surface layer characteristics like bumps or cracks) is determined:

$$RGB(\mathbf{x}) = (1 - C(\mathbf{x}))RGB_{in}(\mathbf{x}) + C(\mathbf{x})RGB_S$$
$$H'(\mathbf{x}) = H(\mathbf{x}) + C(\mathbf{x})B_S D_S(\mathbf{x}) \tag{2}$$

parameter	description
B_S	terrain roughness, achieved by adding elevation displacement, where the surface layer contributes
RGB_S	surface color of surface material
$N(\mathbf{x})$	noise function with codomain [-1;1] used to modify the result of the constraint evaluation multiplicatively
s_S, b_S	scale and bias of the noise function before constraint modification
$h_A(x)$, $h_S(x)$	hat functions for constraint evaluation, where the input is elevation or slope (computed from elevation)

Table 1: Each surface layer provides this set of attributes.

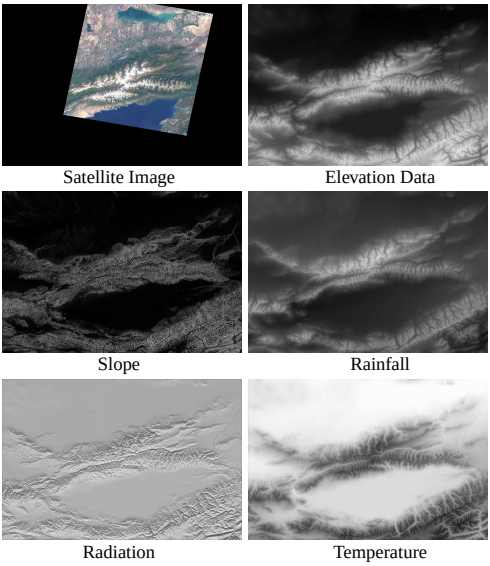


Figure 3: The surface layer constraints and properties are estimated from the real-world input data for the terrain. Please note that for water surfaces the original elevation data does not capture the sea bottom – elevation data is modified there algorithmically. The Landsat satellite image was taken in August 1999 showing little snow cover and snow-free glacier tongues.

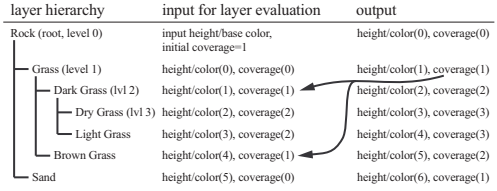


Figure 4: Traversal of a surface layer hierarchy: height data always passes to the layer that is evaluated next, but coverage only feeds back to the layer's children.

5 Mapping the Real World

In this section we describe a method to estimate surface layer constraints and properties for the procedural texturing method from real-world data, i.e. satellite color images, elevation and further morphometric and climatic data. For this, we need registered input data as shown in Fig. 3. Once a surface layer description is constructed from real-world data, we can render arbitrary views of the terrain or images from other terrain parts for which elevation and (simulated) geographic data exists.

5.1 Acquisition of Input Data

First of all, we describe how the input data in our example has been acquired. Computations may vary in detail as specialized models exist for simulating climatic conditions in different geographic regions; the utilization of the resulting data for our texturing method, however, always remains the same.

To regionalize the mean annual air temperature and the precipitation (Fig. 3) in a first step we divided the study area into two parts for temperature and five parts for precipitation, in order to obtain homogeneous areas with similar climatic conditions, e.g. the foothills and the northern slopes the first mountain ridge and the intramountainous basin of lake Issyk-Kul. In a second step, we computed vertical gradients for each area based on a regression model using the existing climate stations in order to get specific values about the temperature decrease and precipitation increase with the altitude. For the Issyk-Kul basin, a significant precipitation increase from west to east occurs, because the air masses obtain humidity while flowing over the large water body. Therefore here a horizontal gradient was also calculated. Then the temperature and precipitation conditions were regionalized using the digital elevation model from the Shuttle Radar Topography Mission (SRTM, resolution 90 m) for each area separately and finally merged using ArcGIS 9.0. The results for the temperature were promising, but not entirely so for the precipitation. In order to integrate the important luv and lee effects of the precipitation, this model was combined with [BL05] for High Asia, which includes these effects, based on the calculated main wind direction using a circulation model. However this model has a lower resolution and integrates only three climate stations in northern Tien Shan and Issyk-Kul basin. The radiation has a high impact on the surface conditions, e.g. the vegetation belts and the snow line are shifted to higher altitudes than on the northern slopes due to the dryness on the southern slopes caused by high radiation input. Therefore the potential direct solar radiation (the direct radiation which the earth surface would receive in cloudy conditions) was modeled, in order to obtain its differentiation. This model has an iterative approach taking the astronomic and atmospheric standard parameters (such as the geographic position or the atmospheric extinction) into account. For more information we refer the interested reader to [BL05, TH04, Bol06].

5.2 Acquiring Surface Layer Descriptions

Several processing steps are involved to acquire the surface layer description. These are presented in chronological order in the following sections. To capture the complexity of nature, we do not represent the surface layer constraints by simple hat functions, but by arbitrary constraint functions. These functions are acquired by sampling the real-world data and stored for a later evaluation. Our texturing parameter estimation has to perform the following steps:

- Identification and classification of surface materials visible in the satellite image.
- Evaluation under which conditions a class of surface material appears, i.e. for example, which elevation, slope, and temperature have to predominate at a certain location that the terrain is covered by snow and ice.
- Replace previously simple hat functions of the procedural model by sampled constraint functions obtained from the evaluation step.

5.2.1 Surface Type Classification

Although multi-spectral satellite images provide much more information than just RGB color, e.g. infrared channels (in fact often the RGB representations are computed from other frequency bands) we restrict our experiments to this limited data. By this, we can also use data from aerial photography which most often does not offer additional data channels. First, the surface types in the RGB color satellite image are to be classified and each surface type, e.g. rock, sand or grass, will initially be represented by a single surface layer. Premoze et al.[PTS99] present a method for removing shading from aerial ortho-images and classifying surface cover considering surface orientation and application of a maximum likelihood Bayer classifier. However, we found that for our purpose and for rendering distant views we can achieve a satisfying classification more easily. The underlying idea is to transform the color information from RGB into color spaces, that allow an easy, yet sufficient, classification of surfaces. Experiments showed that color models providing luminance and two chromaticity values, like the CIE XYZ, YIQ or YUV models, are not practical, as the color separation is not sufficient. On the other hand, the hue-saturation-value (HSV) model in combination with the original RGB representation proved to work very well. Figure 5 shows the satellite image converted into HSV color space and Figure 7 shows the respective histogram of hue and saturation values as a density plot. Please note, that grayish colors, when converted from RGB to HSV, may end up in undefined hue values due to color quantization. For this, (near) gray colors are identified in RGB space and their hue value is set to a user-defined value (here it is set to the hue of purple colors that do not originally appear in the satellite image). The resulting hue-saturation histogram is well suitable to classify the terrain surface into rock, vegetation, water and snow, and ignores shading due to surface orientation. In our experiments we defined the surface clusters manually by dividing the histogram into four subsets. This step does not require geographic knowledge – the clusters are simply chosen according to the hue-saturation histogram. The application of automatic clustering methods, e.g. learning vector quantization or self-organizing maps (see Kohonen [Koh01] for details), remains subject to future research.

5.2.2 Surface Layer Hierarchy

Building a meaningful surface layer hierarchy automatically from the input data cannot be easily accomplished. This is because the input satellite image, does not provide information on the constitution of the terrain surface, from which a meaningful surface layer hierarchy can be derived – only color values are provided. The evaluation of the layer description, however, also depends on the topology of the layer hierarchy.

Considering these problems, we suggest to use a simple heuristic which imposes certain requirements on the constraint functions described in the next section. Initially we start with a root node and assign the color of the most widespread surface type to it. Then we create a level-1 node for each surface type classified in the previous step. All of these sibling nodes are children of the root node and we assign the average color of each surface type class to the nodes. Parameters for the texture diversification for close views through noise functions remain user-defined. Please note, that colors taken from the satellite image underlie atmospheric scattering and thus colors exhibit a shift to blueish color tones.

5.2.3 Estimating Distribution Functions

The last processing step is the determination of conditions which have to predominate that a certain surface type appears in the texture. In Section 4 these conditions were user-defined by few intuitive quantities and simple hat functions have been used

for evaluation. In real-world, however, such simple functions will not suffice to capture the complex distribution of surface materials. Our goal is to acquire constraint functions from the input data automatically which allow us to reproduce the input satellite image. We compute these functions $F_i(\mathbf{x}) \in [0;1)$ (analogous to Eq. 1) for each input quantity (height, slope, rainfall, temperature and radiation) to obtain $P(\mathbf{x}) = \prod_{i=0}^{n-1} F_i(\mathbf{x})$.

This product of constraint functions $P(\mathbf{x})$ is the main criteria for the appearance of a surface layer and can be considered as amount of contribution to the texture color. However, surface layers are evaluated sequentially (due to the simplified layer hierarchy, see Section 5.2.2) and one surface layer may overwrite the contribution of its predecessor.

Consequently, the problem is to acquire $F_i(\mathbf{x})$ for each surface layer from the respective input quantities such that $P(\mathbf{x})$ approaches 1 if, and only if, the layer is visible at surface location $\mathbf{x}$. Under the assumption, that the surface color can be predicted from the input quantities, we propose to use the following heuristic, which proved, judged subjectively, to produce good results. Please note, that we created one surface layer per surface type (as identified in the clustering step) and we estimate all $F_i(\mathbf{x})$ independently from each other. At first, we scale all input quantities such that their domain is $[0;1)$. Next we compute a histogram for each input quantity which denotes how many pixels exhibit the surface type for a certain input value. For example, how many pixels of the satellite image belong to the vegetation cluster and appear at a certain terrain elevation. We found empirically that appropriate distribution functions can be obtained by scaling (and clamping) the histogram such that its mean value equals to 0.5. Resulting functions for the vegetation layer are shown in Fig. 6. The distribution functions are stored as textures for use in the procedural texturing algorithm. The evaluation the analytic functions is replaced by sampling textures.

A diversification of the landscape coloring is achieved by adding more surface layers. To do that without complicating the constraint function estimation, we add child nodes to those level-1 nodes which represent larger clusters in the hue-saturation histogram (Fig. 7), e.g. the greenish vegetation cluster. These nodes are created with slightly, randomly modified attributes for noise diversification and colors from within the surface type cluster.

Due to the lack of a meaningful layer hierarchy, the estimated constraint functions appear rather complicated. If a layer hierarchy based on geological observations would be available, then the histograms would be less disconnected and smoother. For example, assume a fertile soil type appearing at all heights. A vegetation layer on top of it may appear only at a certain height interval (due to other constraints for example). As a consequence, a simple nearly constant height distribution function for the fertile soil becomes disconnected. This is because the non-existent hierarchy (just one level for the basic distribution functions) does not represent the fact, that the soil layer is the precondition for the vegetation layer. This problem cannot be solved with a simple heuristic and rather requires an interactive approach.

6 Results and Conclusions

We found that our rather simple method provides a plausible and visually pleasing reproduction of natural terrain textures. For estimating the constraint functions we excluded the terrain parts that were cultivated by humans (especially the area north of the mountain range in Fig. 3, where irrigated farmland exists). It is obvious, that a method like ours cannot reproduce such regions: human cultivation does not follow the same rules as natural distribution of plants and soil. One remaining problem with satellite images is cloud cover. Due to the white colors of clouds they can misleadingly classified as snow cover. At least for images exhibiting sparse clouds this problem can be solved with a simple special treatment of the snow surface type: for most terrain regions a minimal altitude for the appearance of snow can be indicated and all pixels that are classified as snow (due to their color), but do not meet this height criterion are discarded. By integrating shortwave infrared images (e.g. Landsat, Channel 7), snow and ice can be distinguished from clouds.

The results achieved with our method are illustrated in Figure 6 showing the same region as visible in the satellite image (Fig. 3). The sea and atmospheric conditions were rendered using appropriate analytic models [PSS99, PA00].

During rendering textures with higher resolution than the input data are generated and the texture diversification due to the noise functions produces at least plausible results without changing the overall appearance of the terrain. For texture generation, we only store the distribution functions and the geographic input data. Each distribution function is

represented by 256 equidistant samples stored as an 8-bit texture. The procedural model is implemented as vertex and fragment shaders and entirely executed on graphics hardware. Rainfall, temperature and radiation is stored in a 1024^2 RGB 8-bit texture, the elevation data for our example is 2048^2 with 16-bit precision. Due to tiling and use of a texture atlas, the resolution of the synthesized texture adapts to the view point. The tiled, adaptive texture (occupying 40mb of video memory) used in Fig. 6 corresponds to a 32768^2 fixed-resolution texture. In general, using textures for terrain rendering is only suitable for distant views – for close view, ground detail like shrubs and rocks have to be represented by geometric objects. The execution time for the two automated steps of the fitting process, computing the hue-saturation histogram and surface classification with constraint function estimation, was 2.0sec and 0.7sec on a P4, 3.4GHz processor for a satellite image resolution of 2400×2200 pixels.

Basically a once fitted procedural model, represents a specific clime zone and soil constitution. Thus, changes to the height-field can be made and after redoing the geographic simulations or extrapolating measured data, images of the modified terrain can be rendered with likely realistic texturing.

In the future we would like to exploit more information available from satellite imagery, e.g. infrared bands, to ensure a more detailed and accurate classification. To fully automate this process, appropriate clustering methods have to be applied and validated. Identification of surface layer dependencies and constructing a layer hierarchy accordingly is also subject to research.

Acknowledgement: this project was funded by the DFG project *Interaktive Visualisierung prozeduraler Modelle*.

References

[BL05] BÖHNER J., LEHMKUHL F.: Environmental Change Modelling for Central and High Asia: Pleistocene, Present and Future Scenarios. In *Boreas 34(2)* (2005), pp. 220–231.

[Blo98] BLOW J.: Implementing a Texture Caching System. *Game Developers Magazine* (April 1998).

[Blo00a] BLOOM C.: Terrain Texture Compositing by Blending in the Frame-Buffer. *Available online at* `http://www.cbloom.com/3d/techdocs/splatting.txt` (2000).

[Blo00b] BLOW J.: Terrain Rendering at High Levels of Detail. *Available online at* `http://number-none.com/blow/papers/terrain_rendering.pdf` (2000).

[Bol06] BOLCH T.: Climate Change and Glacier Retreat in Northern Tien Shan (Kazakhstan/Kyrgyzstan) using Remote Sensing Data. In *Global and Planetary Change* (2006).

[CCDH05] COLDITZ C., COCONU L., DEUSSEN O., HEGE H.: Real-time Rendering of Complex Photorealistic Landscapes Using Hybrid Level-of-Detail Approaches. In *Real-time visualization and participation* (June 2005).

[CSHD03] COHEN M. F., SHADE J., HILLER S., DEUSSEN O.: Wang tiles for image and texture generation. *ACM Transactions on Graphics 22*, 3 (2003), pp. 287–294.

[DCSD02] DEUSSEN O., COLDITZ C., STAMMINGER M., DRETTAKIS G.: Interactive visualization of complex plant ecosystems. In *VIS '02: Proceedings of the conference on Visualization '02*, pp. 219–226.

[DS05] DACHSBACHER C., STAMMINGER M.: Cached Procedural Textures for Terrain Rendering. In *Shader X^4* (2005), Charles River Media.

[Fai05] FAIRCLOUGH M.: Terragen v0.9 (Software). *Available online at* `http://www.planetside.co.uk` (2005).

[Gla04] GLANVILLE R. S.: Texture Bombing. In *GPU Gems: Programming Techniques, Tips and Tricks for Real-Time Graphics* (2004), Addison Wesley Professional.

[Koh01] KOHONEN T.: *Self-Organizing Maps*, 3rd ed. Springer, Berlin, Germany, 2001.

[MH05] MCKNIGHT T. L., HESS D.: *Physical Geography: A Landscape Appreciation, 8th edition*. Pearson Prentice Hall, San Francisco, CA, USA, 2005.

[Mic02] MICROSOFT CORPORATION: DirectX 9.0 SDK, 2002, November 2002.

[LH05] LEFEBVRE, S., HOPPE H.: Parallel controllable texture synthesis. In *ACM Transactions on Graphics 24*, 3 (2005), pp. 777–786.

[PA00] PREMOZE S., ASHIKHMIN M.: Rendering Natural Waters. In *PG '00: Proceedings of the 8th Pacific Conference on Computer Graphics and Applications* (Washington, DC, USA, 2000), IEEE Computer Society, p. 23.

[Pid05] PIDWIRNY M.: Fundamentals of Physical Geography - Online Textbook. *Available online at* `http://www.physicalgeography.net` (2005).

[PSS99] PREETHAM A. J., SHIRLEY P., SMITS B.: A practical analytic model for daylight. In *SIGGRAPH '99: Proceedings of the 26th annual conference on Computer graphics and interactive techniques* (New York, NY, USA, 1999), ACM Press/Addison-Wesley Publishing Co., pp. 91–100.

[PTS99] PREMOZE S., THOMPSON W. B., SHIRLEY P.: Geospecific Rendering of Alpine Terrain. In *Rendering Techniques* (1999), pp. 107–118.

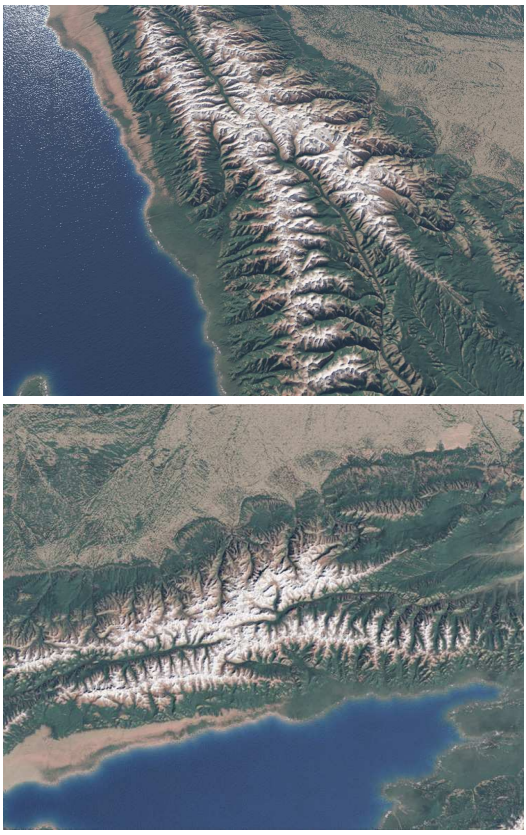
[SGI] SGI OPENGL PERFORMER:. Using Clip Textures. Available online at `http://www.sgi.com/products/software/performer/whitepapers.html`.

[Sti99] STICKLER G.: Solar Radiation and the Earth System. *Available online at* `http://edmall.gsfc.nasa.gov/inv99Project.Site/Pages/science-briefs/ed-stickler/ed-irradiance.html` (1999).

[TH04] THOMAS A., HERZFELD U.: Regeotop: New Climatic Data Fields for East Asia based on localized Relief Information and geostatistical Methods. In *International Journal of Climatology 24* (2004), pp. 1283–1360.

[Ulr02] ULRICH T.: 'Super-size it! Scaling up to Massive Virtual Worlds' course at SIGGRAPH 2002. *Available online at* `http://tulrich.com/geekstuff/chunklod.html` (2002).

[Wlo05] WLOKA M.: Improved Batching via Texture Atlases. In *Shader X3: Advanced Rendering with DirectX and OpenGL* (2005), Charles River Media, pp. 155–167.



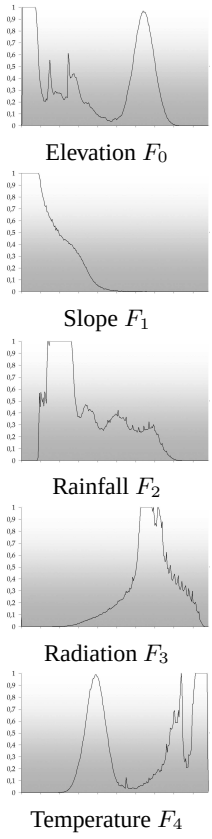


Figure 6: Left: two real-time renderings of the Kazakhstan region with procedural texturing parameters acquired from real-world data. Please note, that the northern lakes and human cultivated regions were not considered during classification and parameter estimation and thus, as a consequence, do not appear in the rendering. Right: the distribution histograms (y-axis: probability) of the vegetation surface type for the different input quantities (x-axis, scaled/biased to domain [0;1)).

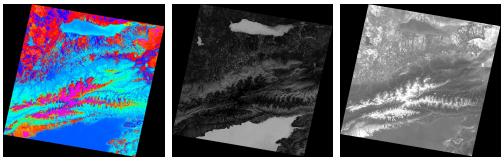


Figure 5: The RGB satellite image is converted into HSV color space. Using these two color spaces allows a feasible clustering of pixels to surface types.

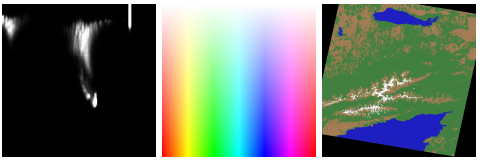


Figure 7: The hue-saturation histogram of the satellite image (which special treatment of gray colors) and the respective color table. The right image shows the classification of surface types for four clusters: water (blue), vegetation (green), rock (brown) and snow (white).