CEIG - Spanish Computer Graphics Conference (2012)
Isabel Navazo and Gustavo Patow (Editors)

Interactive generation of (paleontological) scientific illustrations from 3D-models

S. Schäfer[†], C. Knopp[‡] and D. Krömker[§]

Goethe-Universität Frankfurt, Germany

Figure 1: *A Leptictidium nesutum tooth (left upper M^2). Layers: illumination, outline, period and irregular stippling; result.*

Abstract

Scientific illustrations play an important role in the paleontological domain and are complex illustrations that are manually drawn. The artist creates illustrations that feature common expressive painting techniques: outlines, both irregular and periodic stippling as well as an abstract shaded surface in the background. We present a semi-automatic tool to generate these illustrations from 3D-models in real-time. It is based on an extensible GPU-based pipeline to interactively render characteristic into image-layers that are combined in an image-editing fashion. The user can choose the techniques used to render each layer and manipulate its key aspects. Using 3D- and 2D-painting the artist can still interact with the result and adjust it to his or her liking.

Categories and Subject Descriptors (according to ACM CCS): I.3.3 [Computer Graphics]: Picture/Image Generation—Display algorithms

1. Introduction

An important part of paleontological research is the publication of new findings and insights gained through close examination of specimens. To better illustrate key-characteristics these specimens are often depicted with a scientific illustration that is manually created in a tedious and time-consuming task. The images shown in articles consist of outlines, distinctive types of stippling and abstract surface shading into a single image, as can be seen in figure 2.

While the computer-assisted creation of certain aspects of these illustrations are well researched in the Non-Photorealistic-Rendering community it is the combination of these techniques paired with the desire to give the result an individual touch that has prevented a software tool to be successfully used for the creation of the images. It is clear that there is a high demand for such a tool in the paleontological community, but it is also imperative that the user has the possibility to manually interact with the result and change it to his or her liking. To our knowledge no such tool exists.

In this paper we introduce a novel extensible GPU-based layer approach to create scientific illustrations from 3D-models by rendering it with several well proven Non-Photorealistic-Rendering algorithms. Our contribution is the use of layers with a user definable content to assemble the scientific illustration. The input for a layer is either a painted or calculated texture and it is rendered "as is" or with a

† e-mail: schaefer@gdv.cs.uni-frankfurt.de
‡ e-mail: chknopp@stud.uni-frankfurt.de
§ e-mail: kroemker@gdv.cs.uni-frankfurt.de

DOI: 10.2312/LocalChapterEvents/CEIG/CEIG12/109-112

selected NPR-algorithm. The layers can be created and arranged to the liking of the user. The process is furthermore enhanced by allowing an artist to sculpt the model. By using a 3D-model as an input we can not only use 3D-algorithms to extract features but also allow the user to freely position both model and camera and choose an illumination scenario instead of using a fixed 2D image. The tool is created with paleontological scientific illustrations in mind but can theoretically be used in other areas.

In the next chapter we examine a representative scientific illustration from a paleontological fossil that serves as an archetype and template for our approach. The third chapter discusses related work and techniques used. An overview of our approach is presented in chapter four followed by a detailed description on how we realized the concept. The last two chapters discuss the results we created with our tool and talk about the next steps.

2. Requirements review

In figure 2 a paleontological scientific illustration is shown. The first thing one can notice is the complexity of the illustration:

1. *Outlines* are used to mark the silhouette and to separate overlapping fossil parts. These lines are often idealized from a concrete object.
2. *Irregular stippling* points mark dark areas. These are created not only by a local illumination but also by ambient occlusion. The chaotic nature of these points help to perceive an area as dark without giving any clue of structure or surface texture.
3. *Periodic stippling* points on the other hand are used to explicitly mark surface patterns or important curvatures. They are often following an idealized impression of the object. These abstractions are easy to see for the trained eye but not trivial to be detected by an algorithm. While a real human can not draw a *true* periodic stippling the illustration shows a similar characteristic.
4. In the background the *illuminated surface* is subtly blended in. The illumination helps a viewer to better estimate the curvature of the fossil and recognize the model as a three-dimensional object and not just a flat surface.

3. Related work

The global pipeline of reconstructing a surface model from an imagestack and then render it using several NPR techniques was described in the poster [SH11]. This paper is a continuation of the research and extends the rendering algorithm with the introduction of layers and painters. In the original work the final image was created by simply overlaying a stippled image with an outline image. Our approach by using general user configurable layers is much more open and allows for all kinds of different image effects. It is also possible to use other NPR-algorithms besides stippling to fill layers. A second difference is the mentioned integration of painters to sculpt the model or to paint onto surface textures.

The 3D-models of the paleontological fossils used throughout this paper are a surface reconstruction from a motorfocus microscope image-stack using the method described in [NS10] and [SHK10].

All expressive painting techniques are well-researched NPR techniques from which we present the ones we implemented in our pipeline.

A good overview for view dependent outlines and contour lines can be found in the siggraph course *Line Drawings from 3D models* [RCDF08] in the *suggestive contours* section. A technique presented by Luft et al. [LCD06] called *Image enhancement by unsharp masking the depth buffer* is very efficient in rendering smooth outlines from depth differences in the image.

One notable stippling technique called *Recursive Wang tiles for real-time blue noise* [KCODL06] uses precalculated blue-noise tiles to create a real-time recursive stippling with very little visible and measurable repeating. Another approach from Schmaltz et al. titled *Fast Electrostatic Halftoning* [GSWT11] uses a physical electrostatic model to distribute the particles. The distribution can be calculated as a typical n-body problem that is solved on the GPU in time $\mathcal{O}(n \cdot log\, n)$. The initially regularly placed particles can be jittering to pseudo-random positions in an additional step.

Painting on a surface in 3D is known as *texture painting* and recent publications i.e. [Dro07] or [GeW11] have shown that this can be done entirely on the GPU

A good overview of realtime anti-aliasing solutions can be found in [JGY*11]. We choose Lottes's FXAA [Lot] that produces good results at interactive speeds.

4. Overview

The complete workflow with our tool is as follows:

1. Load model: mesh, ct-scan or imagestack
2. Adjust lighting and position camera and model.
3. Setup layers: select characteristics and how to show it.
4. Fine-tune the parameters or use painter.
5. Store the setup and export a screenshot.

First we renders the object into G-Buffer textures [ST91] in a single renderpass using multiple render targets. With these textures as a starting point all other algorithms can be realized as image space fragment shaders. Once all user chosen layers are rendered they are drawn with alpha blending into a final texture that is then displayed with an applied FXAA filter to reduce visible aliasing.

The basic characteristics (= input for a layer) include illumination, surface structure (like object id or height), outlines (obtained from various techniques) or a manually painted

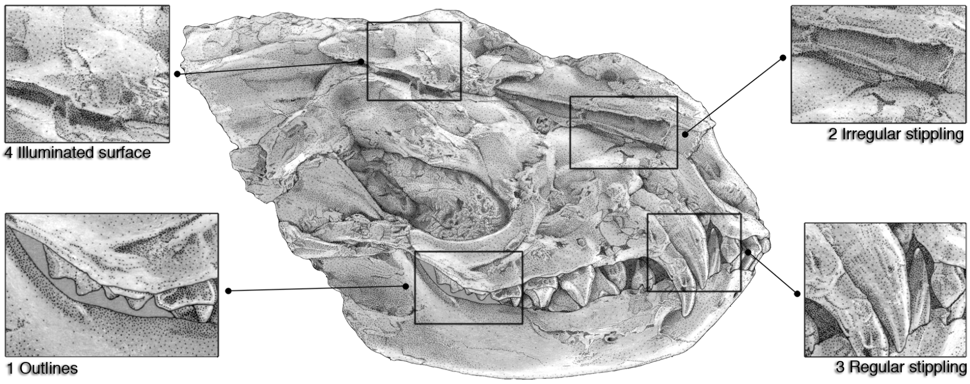

Figure 2: *A fossilized kopidodon head, drawn from a paleontological scientific illustrator (Juliane Eberhardt).*

surface textures. For every layer the user can adjust the tone-curve by specifying brightness-, contrast- and gamma-value and choose a transparency value. The content of the layer can be displayed directly, screen-space hatching or using either period or chaotic stippling.

It is clear that the abstractions and idealizations can hardly be made by algorithms alone and demand an easy way for the user to interact and manipulate with the result. We propose to use a 2D- and a 3D-painter to ensure artistic freedom. With the 3D-painter it is possible to sculpt the model into the abstracted form an artist imagines. The 2D-painter can be used to add and correct automatic extractions (e.g. wrong outlines) as well as manipulating the result of rendering algorithms e.g. add, move or remove stippling points.

5. Method

5.1. Illumination and outlines

We add a screen-space-ambient-occlusion to darken shadowed image areas [BS08] on top off a modified lambertian function to illuminate the model:

$$\text{illumination} = \text{clamp}\left(\frac{(\mathbf{n}\cdot\mathbf{l})^{\gamma} - \text{dark}}{1 - \text{dark}}\right)$$

A simple yet effective method for extracting the outlines is the aforementioned *image enhancement* method. Alternatively an edge-detection filter can be applied to depth- and normal-map.

5.2. Irregular stippling

Irregular stippling is used for dark image parts. These are generated by illumination, shadowing and ambient occlusion passes. We choose to port the *recursive wang tile blue noise* to the GPU as the results are very close to our demand due to the blue noise property. All points are loaded into a VBO and uploaded to the GPU. A vertex shader performs a texture lookup to compare the brightness under the position with the rank (stored in the z-coordinate of a stipple point vertex) to decide if the point is either drawn or discarded. The point density (= perceived intensity) can be changed at run-time.

5.3. Periodic stippling

The *electrostatic halftoning* method uses a physical model to distribute points in the view plane where equally charged points are positioned with equal distance to each other over areas with same intensity. For this reason we chose to use a similar method to render out all curvature lines. Those are mainly drawn by an artist onto a curvature surface texture and are sparsely compared to the whole image. This leads to a small point count that allows us the evaluate the n-body problem at interactive speeds.

5.4. Painting

The painting is done either onto a screen-aligned texture on the view plane or on a model surface texture. An artist can paint pixel-based using mouse strokes or vector-based by placing and modifying lines and curves on the view plane. The paint function itself is realized completely on the GPU using OpenGLs *generalized buffer system*.

The texture coordinates (either from the model or a screen-aligned quad) under the covered areas (either mouse cursor or vector curve) are drawn to a pixel-buffer-object (PBO). The PBO is converted into a vertex-buffer-object (VBO) such that each pixel is interpreted as a vertex with the *xyz*-position stored in the *rgb*-channel. The vertex is now positioned over the target position in texture coordinates and can be rendered into the texture.

5.5. Layer mixing and presets

Every layer is rendered into a final rendertarget using the user-specified tone-curve and transparency value. The final image is post processed by a FXAA anti-aliasing shader.

All parameters can be stored into presets so they can be reloaded and modified later on. They can also be exchanged with the model and painted textures so that other researchers can view the same illustration in 3D.

6. Results

In figure 1 a *Leptictidium nesutum* tooth left upper M^2 is shown. The layers contain the illumination, outline, irregular and periodic stippling. To emphasize the different parts of the tooth the curvature map is used. The outlines help to differentiate overlapping parts of the tooth. The background of a layer is given a blue tint so the content stands out.

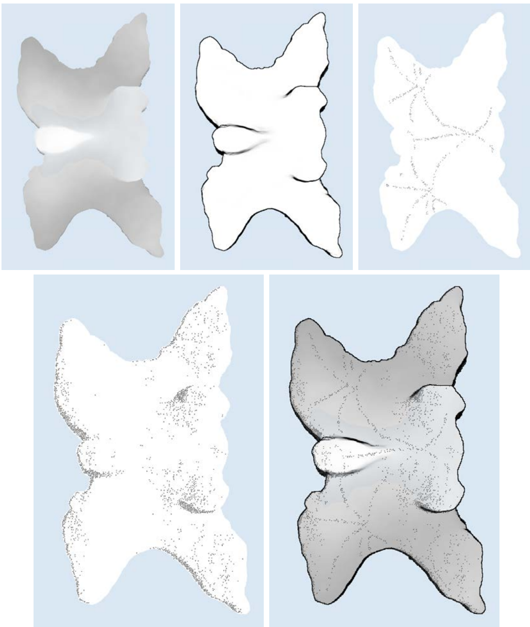

Figure 3: *An ogmorphis vertebra. Layers: illumination, outline, periodic and irregular stippling layer; result.*

The second fossil shown is an *ogmorphis vertebra* in figure 3. The outlines are created using a sobel filter applied to the depthmap with a low threshold setting. The illumination is smoothed and modulated with a light- to dark-gray two-tone shading. The irregular stippling source is the illumination and some added ambient occlusion. The periodic stippling emphasizes the structure of the vertebra bone.

7. Conclusion and Future work

We have presented a method to semi-automatically create paleontological scientific illustrations from 3D-models. The images generated with our prototype come close to our reference images and were judged to be useful by the domain experts, but a real evaluation to answer the question "How convincing are the result generated with the program" has to be completed. We plan an online survey where hand-drawn illustrations are compared to images generated by scientific illustrators as well as novice users of the same object. To evaluate the usability of the program a usability study will be conducted soon.

Finally we would like to extend the use from the paleontological domain to other areas where scientific illustrations are used as every domain has developed their own *visual language* when it comes to scientific illustrations.

Acknowledgement

We like to thank Dr. Thomas Lehman, Dr. Krister Smith and Juliane Eberhardt from Senckenberg Research Institute and Natural History Museum Frankfurt for their expert opinions, support and provided materials.

References

[BS08] Bavoil L., Sainz M.: Screen Space Ambient Occlusion. *Image Rochester NY 6*, September (2008), 413–424. 3

[Dro07] Drone S.: Real-time particle systems on the GPU in dynamic environments. 80–96. doi:10.1145/1281500.1281670. 2

[GeW11] Gross er M., Wacker M.: High Performance Interactive Painting On The GPU. 2011. 2

[GSWT11] Gwosdek P., Schmaltz C., Weickert J., Teuber T.: Fast electrostatic halftoning. *Journal of Real-Time Image Processing* (2011), 1–14. 2

[JGY*11] Jimenez J., Gutierrez D., Yang J., Reshetov A., Demoreuille P., Berghoff T., Perthuis C., Yu H., McGuire M., Lottes T., Others: Filtering approaches for real-time anti-aliasing. In *ACM SIGGRAPH 2011 Courses* (2011), ACM, p. 6. 2

[KCODL06] Kopf J., Cohen-Or D., Deussen O., Lischinski D.: Recursive Wang tiles for real-time blue noise. *ACM Transactions on Graphics 25*, 3 (July 2006), 509. 2

[LCD06] Luft T., Colditz C., Deussen O.: Image enhancement by unsharp masking the depth buffer. *ACM Transactions on Graphics 25*, 3 (2006), 1206. 2

[Lot] Lottes T.: NVIDIA FXAA. URL: http://timothylottes.blogspot.com/2011/07/nvidia-fxaa-39-released.html. 2

[NS10] Nagl F., Schäfer S.: 3D-SurReAL - 3D Surface Reconstruction of Arbitrary (Image) Layer. In *Eurographics Workshop on Visual Computing for Biology and Medicine* (2010). 2

[RCDF08] Rusinkiewicz S., Cole F., DeCarlo D., Finkelstein A.: Line drawings from 3D models. In *ACM SIGGRAPH 2008 classes* (New York, New York, USA, 2008), ACM, pp. 39:1—-39:356. doi:10.1145/1401132.1401188. 2

[SH11] Schäfer S., Heep C.: Surface reconstruction and artistic rendering of small paleontological specimens. In *International Symposium on Non-Photorealistic Animation and Rendering* (2011). 2

[SHK10] Schäfer S., Heep C., Krömker D.: 3D-Oberflächen-Rekonstruktion und plastisches Rendern aus Bilderserien. In *16. Workshop Farbbildverarbeitung* (2010). 2

[ST91] Saito T., Takahashi T.: NC machining with G-buffer method. *ACM SIGGRAPH Computer Graphics 25*, 4 (July 1991), 207–216. 2