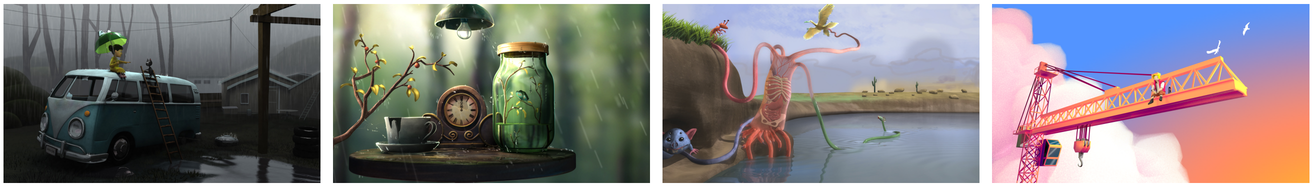
Art Directed Rendering & Shading Using Control Images

Ergun Akleman
Departments of Visualization &
Computer Science and Engineering
Texas A&M University

Siran Liu
Department of Visualization
Texas A&M University

Donald House
School of Computing
Clemson University



(A) (B) (C) (D)

Figure 1: *Example frames from class projects: (A) by Tim Swartz & Jay Jackson after an illustration by Goro Fujita. (B) by Yinan (Yvonne) Xiong after an illustration by Artyakov Artyom. (C) by Chethna Kabeerdoss after an illustration by Rachel Cunningham Wang. (D) by Justin Hollis, after an illustration by Sylvain Sarrailh (aka Tohad).*

In this work, we present a simple mathematical approach to art directed shader development. We have tested this approach over two semesters in an introductory level graduate rendering & shading class at Texas A&M University. The students in the class each chose an artist's style to mimic, and then easily created rendered images strongly resembling that style (see Figures 1). The method provides shader developers an intuitive process, giving them a high level of visual control in the creation of stylized depictions.

two control images shading parameter rendering

Figure 2: *A simple example demonstrating the concept. The first two images are control images, the third image is the shading parameter image and the final image is the weighted average of the control images for every pixel of the shading parameter image.*

In our approach, the shader functions are parametric functions that satisfy the partition of unity, a concept that is widely used in Computer Aided Geometric Design. Parametric functions with this property, such as Bezier or B-spline functions, guarantee that geometric shapes stay inside of the convex hull defined by control vertices, providing an intuitive interaction mechanism for obtaining desired shapes.

SIGGRAPH 2015 Posters, August 09 – 13, 2015, Los Angeles, CA.
ACM 978-1-4503-3632-1/15/08.
http://dx.doi.org/10.1145/2787626.2792612

To develop a specific shader for a given shader function, artists provide a set of texture-mapped control images, which act like control points of parametric functions. With these shader functions, final renderings are guaranteed to be weighted averages of these control images. Figure 2 demonstrates a simple example demonstrating weighted average.

The resulting styles are mainly controlled by the choice of shader functions. We prefer to use non-uniform B-splines since they provide a wide variety of styles. For instance, zero-degree B-splines are appropriate for styles such as cartoon or crosshatching, since they can support discontinuity. Similarly, first-degree B-splines, like in our example figure, support linear color changes with derivative discontinuities. Given this underlying structure, it is easy to provide an intuitive interface that allows the artist choose their style. As a result, the artist's work is simply deciding on a shader function to obtain the desired style and providing control images to obtain the desired color distribution.

Parameters of the shader functions, which we call "shading parameters", are real numbers between 0 and 1. Each shading parameter is an n-tuple, where n is the number of color channels. In practice, our experience has shown that there are only six linearly independent shading parameters that are important for artistic applications: diffuse, specular, silhouette, refraction, Fresnel, and caustic.

In this framework, what is usually considered as shader development goes into computing shading parameters. For instance, the diffuse shading parameter comes from combining direct and indirect contributions of lights in the scene, including shadows and additional effects such as ambient occlusion. This combination turns them into a lump sum n-tuple, that is used as a diffuse parameter.

This approach guarantees that the overall look of resulting 3D renderings are not significantly affected by how the shading parameters are computed. For instance, if shadows are not included in the computation of the diffuse parameter, shadows disappear but the overall look and feel of the rendering does not really change. This predictability is especially useful for artists, enabling them to be able to focus mainly on final appearances.