Charcoal Rendering and Shading with Reflections

Yuxiao Du
Department of Visualization
Texas A&M University

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

Figure 1: *Examples of results from our charcoal shader. As it can be seen from these examples, we can obtain consistent look-and-feel of hand-drawn charcoal drawing effect for a variety of materials and shapes.*

Abstract

In this work, we have developed an approach to include global illumination effects into charcoal drawing (see Figure 1). Our charcoal shader provides a robust computation to obtain charcoal effect for a wide variety of diffuse and specular materials. Our contributions can be summarized as follows: (1) A Barrycentric shader that is based on degree zero B-spline basis functions; (2) A set of hand-drawn charcoal control texture images that naturally provide desired charcoal look-and-feel; and (3) A painter's hierarchy for handling a high number of shading parameters consistent with charcoal drawing.

Keywords: Rendering, Non-Photorealistic Rendering, Charcoal Rendering

Concepts: •**Computing methodologies** → **Non-photorealistic rendering;**

1 Introduction and Motivation

Introduction: During the last two decades many non-photorealistic (NPR) rendering methods are developed to simulate charcoal drawing [Majumder and Gopi 2002; Sousa and Buchanan 1999]. However, there has been no work to attempt global illumination effects such as reflection into charcoal drawing. This is mainly because most of real charcoal drawings do not really demonstrate such effects. On the other hand, some contemporary artists started to incorporate global illumination qualities such as reflection, refraction and caustics while retaining the distinct charcoal drawing identity in their artworks. It is, therefore, possible to analyze such examples to include global illumination effects into charcoal drawing.

General Problem Definition: Development of a rendering and shading framework to obtain a desired look-and-feel is not an easy task in practice. Shade Trees architecture laid the foundations of the procedural shader concept to create any desired look-and-feel [Cook 1984] . Despite the success of shaders and shading languages [Hanrahan and Lawson 1990], to obtain a desired style is still a challenge even for highly qualified lighting technical directors. Recently, a new approach, called Barrycentric Shaders, is developed to simplify shader development through a more intuitive and streamlined process [Akleman et al. 2016]. The method has successfully used in some specific artists' styles including a recent Chinese painter [Liu and Akleman 2015]. In this work, we present a more general application of Barrycentric Shaders that can be used to obtain any style that resemble charcoal drawing.

Motivation: Our goal in this work is to achieve hand drawn charcoal effects in a streamlined process that can be used in a standard rendering pipeline. We, therefore, developed a generic Barrycentric shader that can easily be used in any rendering pipeline to provide charcoal drawing along with global illumination. The most important property of our charcoal shader is that it requires a minimal amount of work to obtain the desired style.

Barycentric Shaders: In Barrycentric shaders, the shading equations are given in the form of

$$C(u,v) = \sum_{i=0}^{M} B_i(t) T_i(u,v)$$

where $C(u,v)$ is rendered color of the point (u,v), $T_i(u,v)$'s are texture images (we call control images) and t is one of the shading parameters such as diffuse parameter, specular parameter, ambient occlusion or shadow and $B_i(t)$'s are basis functions that satisfy partition of the unity property, i.e.

$$\sum_{i=0}^{M} B_i(t) = 1 \quad \text{and} \quad B_i(t) \geq 0.$$

*e-mail:ergun.akleman@gmail.com

SIGGRAPH '16, July 24-28, 2016, Anaheim, CA,
ISBN: 978-1-4503-4282-7/16/07
DOI: http://dx.doi.org/10.1145/2945078.2945110

Using basis functions that satisfy the partition of unity guarantees that regardless of how the shading parameters are computed during rendering, we can always obtain a consistent style [Akleman et al. 2016]. The key decisions, therefore, are to obtain desired styles are the choices of basis functions and control images.

2 Charcoal Rendering

Barycentric Basis Functions for Charcoal drawing : The heart of our approach is degree zero B-splines that is given as

$$N_{i,0}(t) = \begin{cases} 1 & \text{if } u_i \leq u < u_{i+1} \\ 0 & \text{otherwise.} \end{cases}$$

This set of basis functions guarantees that if u is computed between $u_i \leq u < u_{i+1}$, we choose the color from control image $T_i(u, v)$. If the average color of control image $T_i(u, v)$ is between between u_i and u_{i+1}, this guarantees to obtain a clean texture that provides approximately desired color value.

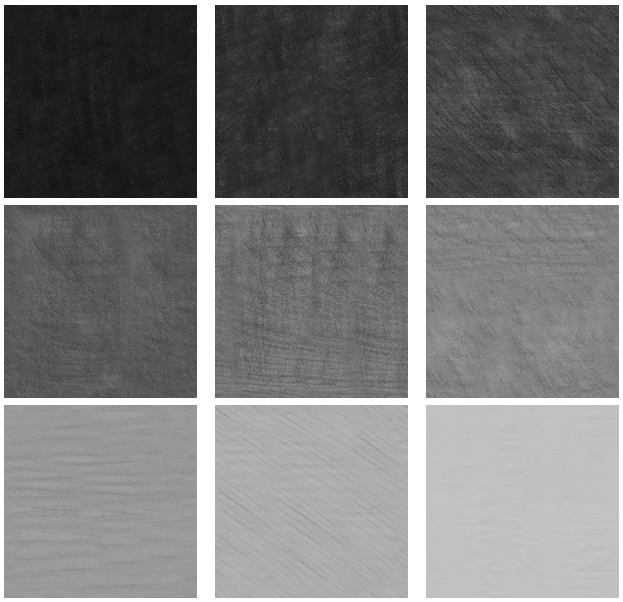

Figure 2: *Control images that we used in Barrycentric Charcoal shader.*

Control Images for Charcoal Drawing: Our control images are hand-drawn textures created by an artist. We scanned them and turned them into repeating wallpaper images. We currently use nine images (See Figure 2). These textures are mapped to objects by using tri-planar projection. This projection is simple and partly avoids foreshortening caused by perspective on texture. There may be other texture mapping strategies that may provide better results, but in our applications we did not see any visible problem in wide range of shapes.

Shading Parameters : In our shaders, we use the following six parameters: (t_0) A single lump sum diffuse parameter that provide a combination of all diffuse reflections and ambient occlusion; (t_1) Shadow parameter, (t_2) Border (also called Outline or Silhouette) parameter, (t_3) Reflection parameter, and (t_4) Specular highlight (Phong-like) parameter. An important issue is that the diffuse parameter must not include any hidden shadow information to control shadow independently like an artist . We remove hidden shadow term that corresponds to negative $\cos\theta$ by using $0.5\cos\theta + 0.5$ [Gooch et al. 1998]. Another key issue is the order of which these parameters are combined. That should be implemented exactly like how it is done charcoal drawing to be obtain similar look-and-feel.

Painter's Hierarchy: If we consider all of these parameters in a general parametric function, it is very hard to keep track of control images since there would be a need for too many of them. The solution to this problem comes from the fact that charcoal artists create final results in several stages. We observed that it is possible to reduce complexity of shading functions using a hierarchy similar to charcoal artists' order. Charcoal artists first create a base image that corresponds to the most essential part of the painting and a light outline. This base image and outline is eventually drawn over with diffuse illumination. They later add shadows and redraw outlines. For charcoal artists, reflection and specular highlights come the last. For specular highlights they use white chalk.

Charcoal Drawing Hierarchy: We use the same hierarchy of parameters given in shader parameters. We first use t_0 to compute a base image. Then we combine base image with shadow using t_1 parameter. Then we add outlines using t_2 parameter. Using t_3 parameter, we add mirror reflections. The last term added is specular highlights that is added with t_4 parameter. Since each of these are computed by barycentric formula, namely degree zero B-splines, combination of all these equations is guaranteed to provide partition of unity.

Further Discussion: One property that can further simplify shader is that, except for diffuse, most other parameters do not really require full texture images. Most of them can be described using a single color. For instance, we do not usually have to change the color of a outline along the silhouette boundary and single color is usually sufficient.

References

AKLEMAN, E., HOUSE, D. H., AND LIU, S. 2016. Barycentric shaders: Art directed shading using control images. *Proceedings of Expressive'2016: Computational Aestetics Conference*, Accepted.

COOK, R. L. 1984. Shade trees. *ACM Siggraph Computer Graphics 18*, 3, 223–231.

GOOCH, A., GOOCH, B., SHIRLEY, P., AND COHEN, E. 1998. A non-photorealistic lighting model for automatic technical illustration. In *Proceedings of the 25th annual conference on Computer graphics and interactive techniques*, ACM, 447–452.

HANRAHAN, P., AND LAWSON, J. 1990. A language for shading and lighting calculations. *ACM SIGGRAPH Computer Graphics 24*, 4, 289–298.

LIU, S., AND AKLEMAN, E. 2015. Chinese ink and brush painting with reflections. In *SIGGRAPH 2015: Studio*, ACM, 8.

MAJUMDER, A., AND GOPI, M. 2002. Hardware accelerated real time charcoal rendering. In *Proceedings of the 2nd international symposium on Non-photorealistic animation and rendering*, ACM, 59–66.

SOUSA, M. C., AND BUCHANAN, J. W. 1999. Computer-generated graphite pencil rendering of 3d polygonal models. In *Computer Graphics Forum*, vol. 18, Wiley Online Library, 195–208.