DEU FMD 21(63), 859-867, 2019

Dokuz Eylül Üniversitesi Mühendislik Fakültesi Fen ve Mühendislik Dergisi
Dokuz Eylul University Faculty of Engineering Journal of Science and Engineering

Basılı/Printed ISSN: 1302-9304. Elektronik/Online ISSN: 2547-958X

Real-Time Shading with Phong BRDF Model

Phong BRDF Modeli ile Gerçek Zamanlı Işıklandırma

Murat Kurt [1*]

[1] International Computer Institute, Ege University, Izmir, TURKEY
Sorumlu Yazar / Corresponding Author *: murat.kurt@ege.edu.tr

Geliş Tarihi / Received: 20.08.2018 *DOI:10.21205/deufmd.2019216315*
Kabul Tarihi / Accepted: 26.04.2019 *Araştırma Makalesi/Research Article*

Atıf şekli/ How to cite: KURT, M. (2019). Real-Time Shading with Phong BRDF Model. DEUFMD, 21(63),859-867.

Abstract

In this paper, we propose a novel real-time Bidirectional Reflectance Distribution Function (BRDF) shading interface for creating isotropic BRDFs in image-based lighting. Our proposed BRDF shading interface allows users to generate new BRDFs by tweaking parameters of underlying BRDF model, which is Phong BRDF model. The implementation of our BRDF shading interface utilizes programmable graphics hardware through Shadertoy, and it provides a real-time visualization of the material on an object in environment lighting.

Keywords: *BRDF, Phong BRDF model, Modeling interface, Appearance modeling, Global illumination, Rendering*

Öz

Bu makalede, görüntü tabanlı ışıklandırma altında izotropik İki Yönlü Yansıma Dağılım Fonksiyonları (BRDFs) oluşturmak için gerçek zamanlı yeni bir BRDF ışıklandırma ara yüzü önermekteyiz. Önerdiğimiz BRDF ışıklandırma arayüzü kullanıcıların Phong BRDF modelinin parametrelerini değiştirerek yeni BRDF'ler oluşturmasına müsaade etmektedir. BRDF ışıklandırma arayüzümüzün implementasyonu Shadertoy aracılığıyla programlanabilir grafik donanımından faydalanmaktadır ve çevresel ışıklandırma altında bir objenin üzerinde malzemenin gerçek zamanlı olarak gösterimini sağlamaktadır.

Anahtar Kelimeler: *BRDF, Phong BRDF modeli, Modelleme arayüzü, Görünüm modelleme, Gobal ışıklandırma, örüntüleme*

1. Introduction

The real-time visualization of materials in different environmental lighting conditions is one of the most intensive work areas in computer graphics [1]. Interest in three dimensional (3D) interactive applications is increasing day by day. The various 3D models are used instead of many on-scene images, from computer games to virtual reality works, archival of historical works, to electronic commercials. As a natural consequence of this situation, there is a growing interest for simple yet affordable systems that allow the creation of realistic 3D models together with material properties, such as diffuse color, specular color, gloss, translucency, and texture that interact with each other in a complex way. Despite being expensive in terms of time and labor, editable material design is the most commonly used method in 3D modeling process [2, 3].

Representing homogeneous materials in physically-based rendering systems is generally achieved by the Bidirectional Reflectance Distribution Function, BRDF, introduced by Nicodemus et al. [4]. The BRDF can be represented by analytical models [5, 6, 7] or

data-driven models [8, 9, 10]. Compared to the data-driven BRDF models, the analytical BRDF models are easier to edit, more suitable for real-time rendering applications, but provides less accurate representations. Data-driven BRDF representations [8, 9, 10] need more storage and computation, as they are mainly based on a compact representation of measured BRDF data [11]. Since our goal is to create a lightweight rendering framework for interactive material and lighting design, we select to use a simple analytical BRDF model, which is called as Phong BRDF model [6]. As our other goal is to create a real-time material editing framework that works on various platforms (web, mobile, PC, etc.), and allows to being extendible to a Virtual Reality (VR) application [12], we prefer to use Shadertoy [13] as an implementation platform, which provides these kind of capabilities.

In this paper, we present a new real-time BRDF editing and rendering interface for material and lighting design through direct control of the material's BRDF by changing the parameters of underlying BRDF model, which is analytical Phong BRDF model [6]. The implementation of our BRDF shading interface utilizes programmable graphics hardware through Shadertoy [13], and it provides a real-time visualization of the material on an object in different environment lighting conditions.

The paper is organized as follows: related works are mentioned in Section 2. Next, our proposed BRDF shading interface is explained in Section 3. Section 4 presents the results. Finally, Section 5 concludes the paper.

2. Related Work

Phong [6] presented several shading techniques corresponding to different methods of object modeling and the related hidden surface algorithms. Human visual perception and the fundamental laws of optics were considered in the development of shading rule that provides better quality and increased realism in generated images. Phong also proposed an empirical BRDF model, which is used to represent isotropic materials. Cook and Torrance [14] presented a microfacet-based reflectance model for rendering computer synthesized images. The analytical model accounts for the relative brightness of different materials and light sources in the same scene. It describes the directional distribution of the reflected light and a color shift that occurs as the reflectance changes with incidence angle. They presented a method for obtaining the spectral energy distribution of the light reflected from an object made of a specific real material. In addition, a procedure for accurately reproducing the color associated with the spectral energy distribution was discussed by them. The model was applied to the simulation of a metal and a plastic materials. Ngan et al. [15] have showed that the Cook-Torrance BRDF model provides close fit to real BRDF measurements [11]. Cabral et al. [16] implemented a reflection space Image Based Rendering (IBR) algorithm to radiance environment maps. A radiance environment map pre-integrates a BRDF with a lighting environment. Using the reflection-space IBR algorithm on radiance environment maps authorizes interactive rendering of arbitrary objects with a large class of complex BRDFs in various lighting environments. The ultimate simplicity of the final version of algorithm proposes that it is widely and immediately precious given the ready availability of hardware-assisted environment mapping. McAllister et al. [17] proposed texture maps that contain at each texture element (texel) all the parameters of Lafortune BRDF representation [18] as a compact, but quite general surface appearance representation. They described a method for rendering such surfaces very quickly on available graphics cards and demonstrated the method with real, measured surfaces and hand-painted surfaces. Furthermore, they proposed a method of rendering such spatial BRDFs using prefiltered environment maps. Just one sequence of maps is needed for rendering the different BRDFs.

Colbert et al. [3] proposed a BRDF-Shop, which is an interface for creating physically-correct BRDFs through positioning and manipulating highlights on a spherical canvas. The BRDF-Shop uses an extension of Ward BRDF model [19] for creating new BRDFs. Green et al. [20] presented a technique for approximating isotropic BRDFs and precomputed self-occlusion that enables accurate and efficient prefiltered environment map rendering. Their approach uses a nonlinear approach of the BRDF as a weighted sum of isotropic Gaussian functions. Their representation needs a very few quantities of memory, can completely represent BRDFs of arbitrary sharpness, more than anything, effective to render. They precomputed visibility due to self-occlusion and keep in reserve a low-frequency approach appropriate for glossy reflections. They demonstrated their method by

fitting their representation to measured BRDF data, yielding high visual quality at real-time frame rates. Krivanek and Colbert [21] have proposed an analysis of numerical integration based on the sampling theory of suppression with integration error pre-filtering resulting from the aliasing. They derived a pre-filter for evaluating the illumination integral yielding filtered importance sampling, a simple GPU-based rendering algorithm for image-based lighting. Additionally, they extended the algorithm with real-time visibility computation. Without preliminary calculation, the developed algorithm supports completely dynamic scenes and, most importantly, is easy to apply. Kurt et al. [5] proposed a new physically plausible, anisotropic BRDF model for fitting and for importance sampling in global illumination rendering. They have shown that the proposed model is more successful than the existing BRDF models in terms of accuracy. Effective methods for sampling the proposed anisotropic BRDF model have been tested in the GPU based real-time rendering algorithm [21]. They also showed that the new BRDF model has an effective real-time rendering performance. Bagher et al. [22] proposed a new objective function based on compressive weighting that controls rendering error in high-dynamic-range BRDF fits better than previous factorization approaches. The developed method compactly combines a more comprehensive set of materials than the state-of-the-art parametric approaches. They experimentally validated the benefit of the microfacet model over a naive orthogonal factorization and showed that fidelity for diffuse materials is modestly improved by fitting an unrestricted shadowing/masking factor. They also compared against a recent data-driven factorization approach [23] and showed that their approach improves rendering accuracy for most materials while reducing storage by more than 10 ×. Bagher et al. [22] showed that their proposed BRDF representations are suitable for real-time rendering by implementing them into a WebGL based rendering application.

Holzschuch and Pacanowski [7] presented a two-scale microfacet-based BRDF model, which considers to represent reflection and diffraction effects of measured materials. Indeed, the proposed analytical BRDF model uses an extension of Cook-Torrance BRDF model [14]. Authors showed that the parameters of the proposed BRDF model can be edited to create new BRDFs. Soler et al. [8] devised a manifold that uses a Gaussian Process Latent Variable Model (GPLVM) to represent measured BRDFs [11]. This manifold-based representation allows to efficiently interpolate and extrapolate measured BRDFs. Soler et al. [8] showed that the proposed model can be used in real-time rendering and editing of measured BRDFs by interpolating in the manifold between two BRDFs from the training set. Sun et al. [9] proposed a robust diffuse-specular separation algorithm for editing the diffuse and specular parts of measured BRDFs, separately. The proposed algorithm essentially connects analytic and measured BRDFs by doing diffuse-specular separation. It also leads to a compact and accurate representation of measured BRDFs. Sun et al. [9] showed that diffuse-specular separation allows new types of editing operations for measured BRDFs, such as editing colors, highlight removal, and mixing reflectances of different measured BRDFs. Tongbuasirilai et al. [10] proposed compact, intuitive and separable data-driven BRDF models, which are based on a Projected Deviation Vector (PDV) parameterization and an iterative factorization of measured isotropic BRDFs [11]. It's shown that intuitive BRDF editing can be done by a mixture of PDV factors.

In this work, we focus on the material design by allowing to edit parameters of the Phong BRDF model [6], which provides a wider gamut of possible isotropic BRDFs. We use Shadertoy [24] to implement our lightweight BRDF editing and rendering framework. Shadertoy allows to utilize programmable graphics hardware, and allows developers to live-code procedural shaders that react to music, videos, and webcams by using HTML5 and WebGL [25, 26]. These creations are shared with the community, making Shadertoy a great repository for finding inspiration, learning, and teaching about shading, reactivity, and rendering [24]. Another property of Shadertoy is to allow creating state-of-the-art VR applications, which we consider as an advantage. We provide a real-time graphical user interface (GUI) to visualize objects and create new BRDFs under different environment lighting conditions. Our user-friendly BRDF shading interface can be used on various platforms, such as a standard web page, mobile, and PC. Our material editing framework provides fast rendering times, and can be converted into a VR application with capabilities of Shadertoy.

3. BRDF Shading Interface

Shading model in physically-based rendering systems is generally the Bidirectional Reflectance Distribution Function, BRDF, introduced by Nicodemus et al. [4]. The BRDF is generally denoted as f_r and is a useful mathematical formulation for calculating the reflection of light from a surface. For a given point of p, the BRDF can be calculated as the ratio between reflected outgoing radiance L_o and incoming irradiance E_i using the outgoing direction $\vec{\omega}_o$ and the incoming direction $\vec{\omega}_i$, respectively. In the rest of the paper, we ignore the notation of the point p in the BRDF, as we focus on representing homogeneous materials in which BRDFs are independent of the surface point, p. Thus, the BRDF is expressed as [5]:

$$f_r(\vec{\omega}_i, \vec{\omega}_o) = \frac{dL_o(\vec{\omega}_o)}{dE_i(\vec{\omega}_i)} = \frac{dL_o(\vec{\omega}_o)}{L_i(\vec{\omega}_i)cos\theta_i d\vec{\omega}_i}, \quad (1)$$

where L_i is incoming radiance, θ_i is the angle between $\vec{\omega}_i$ and the surface normal $\vec{n}$. The physically based BRDFs have to satisfy three properties:

For each direction pair $\vec{\omega}_i$ and $\vec{\omega}_o$, the BRDF has to satisfy reciprocity,

$$f_r(\vec{\omega}_i, \vec{\omega}_o) = f_r(\vec{\omega}_o, \vec{\omega}_i). \quad (2)$$

Total energy of the reflected light has to be less than or equal to the energy of incident light, which is called as the energy conservation law. For each direction $\vec{\omega}_o$:

$$\int_{\Omega^+} f_r(\vec{\omega}_i, \vec{\omega}_o)(\vec{\omega}_i \cdot \vec{n}) d\vec{\omega}_i \leq 1, \quad (3)$$

where Ω^+ is the hemisphere of all incoming directions.

The BRDF should be non-negative such that $f_r(\vec{\omega}_i, \vec{\omega}_o) \geq 0$.

Our BRDF rendering and editing framework has two principal goals. First goal is to provide a mechanism for creating BRDFs in a manner that is both artistic and intuitive. Second goal is to support interactive feedback for clearer understanding of the behavior of the created BRDFs. We satisfy both criteria by providing a user friendly interface that requires an extended Phong BRDF model [6, 15] and an efficient mapping of user interaction to parameters of Phong BRDF model. We chose the Phong BRDF model for our BRDF shading and editing framework as it has an intuitive set of parameters that makes mapping of an artist's interaction to isotropic BRDF creation relatively straight forward.

In our work, we use a normalized version of the Phong BRDF model [6, 15]:

$$f_r(\vec{\omega}_i, \vec{\omega}_o) = \frac{k_d}{\pi} + k_s \frac{(n+2)}{2\pi}(\vec{\omega}_o \cdot \vec{\omega}_r)^n, \quad (4)$$

where k_d, k_s are diffuse and specular color coefficients, $\vec{\omega}_r$ is the reflection vector, which is computed as $\vec{\omega}_r = 2(\vec{\omega}_i \cdot \vec{n})\vec{n} - \vec{\omega}_i$, and n is the power coefficient. In our BRDF rendering and editing framework, k_d, k_s and n terms can be edited to create new BRDFs. To satisfy the energy conservation (see Eq. (3)) and the non-negativity properties, $0 \leq k_d \leq 1$, $0 \leq k_s \leq 1$, $0 \leq k_d + k_s \leq 1$ and $n \geq 0$ should be satisfied in Eq. (4). The Phong BRDF model doesn't satisfy the reciprocity property (see Eq. (2)) because of the computation of the reflection vector ($\vec{\omega}_r$). In Eq. (4), the first part of the Phong BRDF model is named as diffuse lobe, the second part of the Phong BRDF is named as specular lobe [15, 5].

To visualize materials on an object in various environment lighting conditions at real-time frame rates, Kajiya's the rendering equation [27] should be implemented efficiently. The rendering equation is formulized as:

$$L_o(\vec{\omega}_o) = \int_{\Omega^+} L_i(\vec{\omega}_i) f_r(\vec{\omega}_i, \vec{\omega}_o)(\vec{\omega}_i \cdot \vec{n}) d\vec{\omega}_i. \quad (5)$$

To be able to implement the rendering equation computationally efficiently, we extended a rendering algorithm [28], which is publicly available in Shadertoy [13, 24]. As can be seen in Figure 1, we firstly prepared blurred versions of environment maps. These blurred versions are prepared based on a standard averaging operation.

As the BRDF can be separated into two parts, namely, diffuse lobe and specular lobe [15, 5], we compute the rendering equation in two parts. To compute the specular part of the rendering equation, we use blurred versions of environment maps (see Figure 1 (bottom row)) and the incoming light vector ($\vec{\omega}_i$). Firstly, we

construct cube maps by using these blurred versions of environment maps. We draw 9 samples for each side of the cube maps, then take an average of these samples to compute specular environment lighting for each side of the cube maps. The positions of these samples are constant. To compute the final specular environment lighting, the computed specular environment lightings for each side of the cube maps are weighted averaged by using the dot products of incoming light vector ($\vec{\omega}_i$) and normal vectors ($\vec{n}$) of each side of the cube maps. Finally, the specular part of the rendering equation is computed by multiplying the final specular environment lighting and the specular lobe in Equation (4).

To compute the diffuse part of the rendering equation, we use a similar procedure as we follow in the computation of the specular part of the rendering equation. We use the constructed cube maps again. We draw 9 samples for each side of the cube maps, then take an average of these samples to compute the reflection environment lighting for each side of the cube maps. To compute the final reflection environment lighting, the computed reflection environment lightings for each side of the cube maps are weighted averaged by using the dot products of the reflection vector ($\vec{\omega}_r$) and normal vectors ($\vec{n}$) of each side of the cube maps. To compute the diffuse environment lighting, this process is repeated by using the normal vector ($\vec{n}$) instead of the reflection vector ($\vec{\omega}_r$). To get the final diffuse environment lighting, a linear interpolation is applied between the final reflection environment lighting and the diffuse environment lighting by using the power coefficient (n). Finally, the diffuse part of the rendering equation is computed by multiplying the final diffuse environment lighting and the diffuse lobe in Equation (4). In this computation, the rendering equation is simulated by summing the diffuse part and specular part.

4. Results

In this work, we used Shadertoy [13, 24] to implement our BRDF shading and editing interface. As Shadertoy is a cross-browser online community and tool for creating and sharing shaders through WebGL, used both for learning and teaching 3D computer graphics in a web browser [13], our BRDF shading interface includes these properties and away from the challenges of only being a desktop application.

Our results were gathered on a PC with Intel Xeon CPU E5-2640 v3 2.60GHz two processors, 80GB RAM and a Nvidia Quadro K4200 4GB graphics card. In this setting, our material editing and rendering framework provides high performance (~60 fps) when rendering dynamic scenes with arbitrary BRDFs given by an analytic Phong BRDF model. Our BRDF rendering algorithm uses only a small number of samples (9 samples for each face of the cube map, a total of 54 samples in practice) for visually and numerically acceptable results, making the implementation suitable to the GPU, and usable in mobile, web browser and VR.

Our BRDF shading and editing interface allows to edit parameters of underlying BRDF model, which is the Phong BRDF model. As it's seen in Figure 2, when the diffuse coefficient (k_d) in Eq. (4) is edited to create new BRDFs, the general appearance of created materials is affected. In Figure 3, it's shown that changing the specular coefficient (k_s) mostly affects the color of the specular highlights of the created BRDFs.

The power coefficient (n) of the Phong BRDF model controls the general shape of the specular highlights of created BRDFs. It's an intuitive parameter. As it's seen in Figure 4, while low values of the power coefficient causes to create diffuse BRDFs, high values of the power coefficient causes to create glossy BRDFs.

Furthermore, our BRDF shading and editing interface allows to change the material properties in real-time and various environment lighting conditions. In Figure 5, we demonstrate the ability to change the environment lighting conditions in real-time and view animated scenes without any precomputation. As we use standard cube maps to store environment maps in our rendering algorithm, changing the environment does not affect the real-time frame rates (FPS) of our BRDF shading interface. Our BRDF shading and editing interface is lightweight and implemented in Shadertoy, which makes it usable at web, mobile, PC, and VR efficiently. The major drawback of our framework is that it does not include a very accurate implementation of the rendering equation [27], which we're planning to improve its accuracy as our future work.

The existing material editing systems [3, 5, 21, 7, 8, 9, 10] provide more accurate representations of both BRDF and the rendering equation, but their implementations into various platforms, such as web, mobile, PC and VR are questionable.

Figure 1. To give Monte Carlo rendering effects [27, 21, 5], we use environment maps (the first row, 256 × 256 images) and their blurred versions (the second row, 64 × 64 images) in our rendering and editing framework. The environment maps are Eucalyptus Grove, St. Peter's Basilica and the Uffizi Gallery in the first, second and third columns, respectively [29]. These environment maps and their blurred versions are readily available in Shadertoy [13]

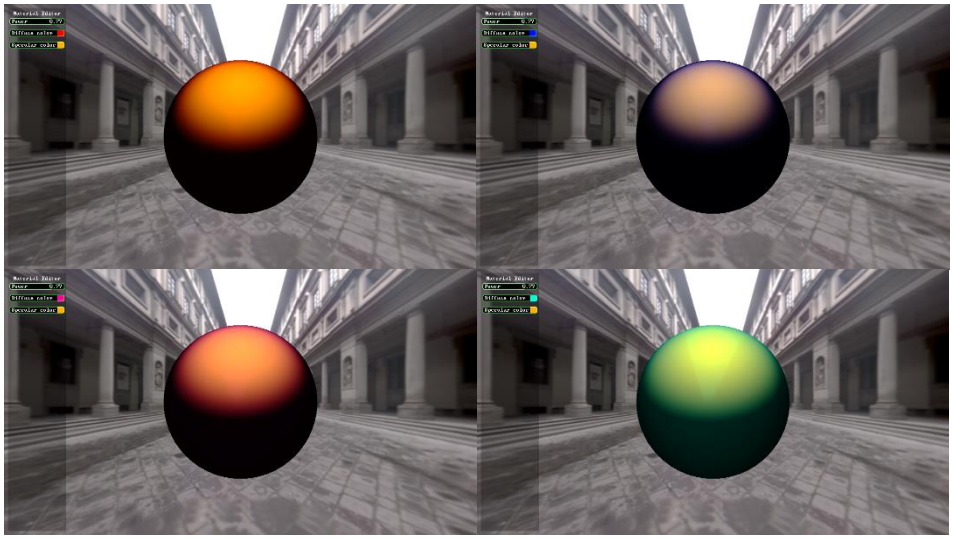


Figure 2. Illustration of the different values of the diffuse coefficient (k_d) and its effect on the Phong BRDF. In this illustration, k_s, n and the environment lighting (the Uffizi Gallery) are constant in all rendered images

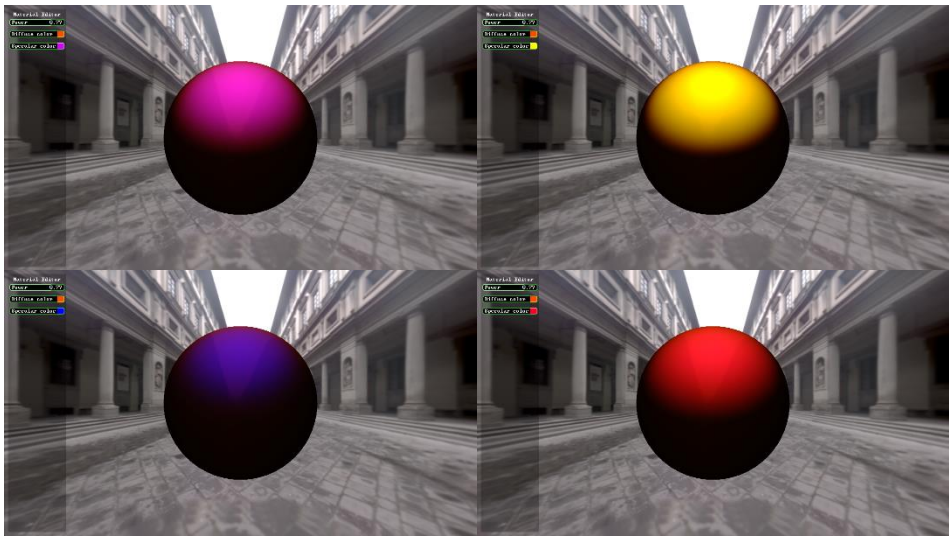
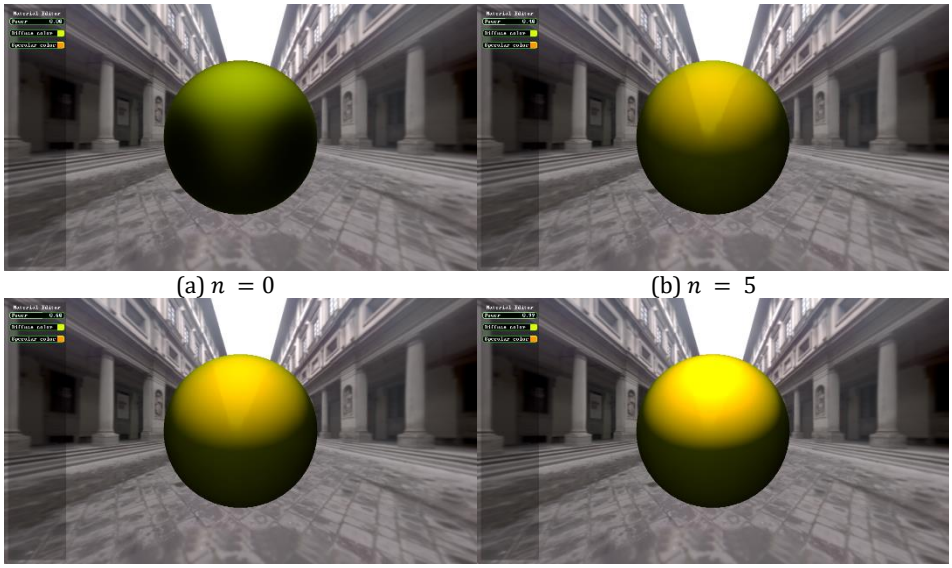


Figure 3. Illustration of the different values of the specular coefficient (k_s) and its effect on the Phong BRDF. In this illustration, k_d, n and the environment lighting (the Uffizi Gallery) are constant in all rendered images



(a) $n = 0$ (b) $n = 5$

(c) $n = 12$ (d) $n = 20$

Figure 4. Illustration of the different values of the power coefficient (n) and its effect on the Phong BRDF. In this illustration, k_d, k_s and the environment lighting (the Uffizi Gallery) are constant in all rendered images. Below each rendered image, we report n values

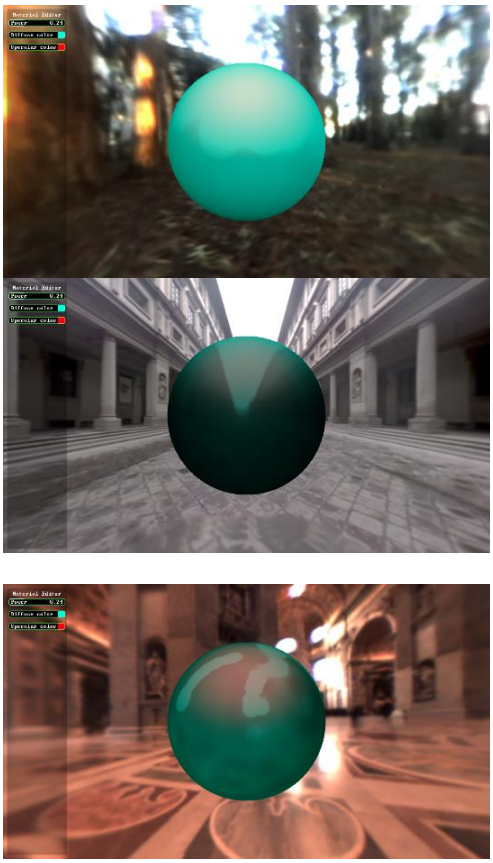


Figure 5. Illustration of the different environment lighting conditions and its effect on the Phong BRDF. In this illustration, k_d, k_s and $n = 4.8$ are constant in all rendered images. From top to bottom: Eucalyptus Grove, the Uffizi Gallery and St. Peter's Basilica are used as environment maps

In particular, VR requires efficient, realistic and cost effective BRDF representations since it poses additional issues, related to the scale versatility, the need of tiling artifact reduction and the computational limitations of mobile devices [12].

5. Conclusion and Future Work

In this study, we presented a BRDF shading and editing interface which includes an efficient GPU-based algorithm for real-time rendering of glossy objects under image-based illumination. As our BRDF shading and editing interface is implemented in Shadertoy [13], it includes all of the properties of Shadertoy, such as ability to work on a standard Web browser, mobile, PC and being extendible as a VR application.

We demonstrated that our BRDF shading and editing interface has ability to change the material properties in real-time and view animated scenes in various environment lighting conditions.

Our BRDF shading and editing interface only allows to create isotropic BRDFs. As a future work, we would like to extend our BRDF shading interface to visualize anisotropic BRDFs in real-time frame rates and in various environment lighting conditions. We're also planning to extend our BRDF shading and editing interface as a VR application, thanks to capabilities of Shadertoy.

Acknowledgments

We would like to thank Paul Debevec for the HDR environment maps used in this work, Mashhuda Glencross for proofreading the paper, and anonymous reviewers for their constructive comments. This work was supported by the Scientific and Technical Research Council of Turkey (Project No: 115E203), the Scientific Research Projects Directorate of Ege University (Project No: 2015/BİL/043).

References

[1] Töral, Ö.A., Ergun, S., Kurt, M., Öztürk, A. 2014. Mobile gpu-based importance sampling. The IEEE 22nd Signal Processing and Communications Applications Conference, IEEE, Trabzon, Turkey, 510–513.

[2] Erdem, M.E., Erdem, I.A., Yılmaz, U., Atalay, V. 2004. Image-based extraction of material reflectance properties of a 3d rigid object. The IEEE 12th Signal Processing and Communications Applications Conference, IEEE, 245–248.

[3] Colbert, M., Pattanaik, S., Krivanek, J. 2006. BRDF-shop: creating physically correct bidirectional reflectance distribution functions, IEEE Computer Graphics and Applications, Vol. 26, No. 1, pp. 30–36.

[4] Nicodemus, F.E., Richmond, J.C., Hsia, J.J., Ginsberg, I.W., Limperis, T. 1977. Geometrical Considerations and Nomenclature for Reflectance. Final Report National Bureau of Standards, Washington, DC. Inst. for Basic Standards, National Bureau of Standards (US).

[5] Kurt, M., Szirmay-Kalos, L., Krivanek, J. 2010. An anisotropic brdf model for fitting and monte carlo rendering, SIGGRAPH Computer Graphics, Vol. 44, No. 1, pp. 1–15.

[6] Phong, B.T. 1975. Illumination for computer generated pictures, Communications of the ACM, Vol. 18, No. 6, pp. 311–317.

[7] Holzschuch, N., Pacanowski, R. 2017. A two-scale microfacet reflectance model combining reflection and diffraction, ACM Transactions on Graphics, Vol. 36, No. 4, pp. 66:1– 66:12.

[8] Soler, C., Subr, K., Nowrouzezahrai, D. 2018. A versatile parameterization for measured material manifolds, Computer Graphics Forum, Vol. 37, No. 2, pp. 135–144.

[9] Sun, T., Jensen, H.W., Ramamoorthi, R. 2018. Connecting measured brdfs to analytic brdfs by data-driven diffuse-specular separation, ACM Transactions on Graphics, Vol. 37, No. 6, pp. 273:1–273:15.

[10] Tongbuasirilai, T., Unger, J., Kronander, J., Kurt, M. 2019. Compact and intuitive data-driven brdf models, The Visual Computer. (Accepted for Publication).

[11] Matusik, W., Pfister, H., Brand, M., McMillan, L. 2003. A data-driven reflectance model, ACM Transactions on Graphics, Vol. 22, No. 3, pp. 759–769.

[12] Guarnera, G.C., Ghosh, A., Hall, I., Glencross, M., Guarnera, D. 2017. Material capture and representation with applications in virtual reality. ACM SIGGRAPH 2017 Courses, ACM, NY, USA, 6:1 – 6:72.

[13] Wikipedia, F.I. 2013. Shadertoy. https://en.wikipedia.org/wiki/Shadertoy (Access Date: 26.04.2019).

[14] Cook, R.L., Torrance, K.E. 1982. A reflectance model for computer graphics, ACM Transactions on Graphics, Vol. 1, No. 1, pp. 7–24.

[15] Ngan, A., Durand, F., Matusik, W. 2005. Experimental analysis of brdf models. The 16th Eurographics Symposium on Rendering, 117–126.

[16] Cabral, B., Olano, M., Nemec, P. 1999. Reflection space image based rendering. The 26th Annual Conference on Computer Graphics and Interactive Techniques, ACM Press/Addison-Wesley Publishing Co., NY, USA, 165–170.

[17] McAllister, D.K., Lastra, A., Heidrich, W. 2002. Efficient rendering of spatial bidirectional reflectance distribution functions. The ACM SIGGRAPH/EUROGRAPHICS Conference on Graphics Hardware, Eurographics Association, 79–88.

[18] Lafortune, E.P., Foo, S.-C., Torrance, K.E., Greenberg, D.P. 1997. Non-linear approximation of reflectance functions. SIGGRAPH '97, ACM Press/Addison-Wesley Publishing Co., NY, USA, 117–126.

[19] Ward, G.J. 1992. Measuring and modeling anisotropic reflection. SIGGRAPH '92, ACM, NY, USA, 265–272.

[20] Green, P., Kautz, J., Durand, F. 2007. Efficient reflectance and visibility approximations for environment map rendering, Computer Graphics Forum, Vol. 26, No. 3, pp. 495–502.

[21] Krivanek, J., Colbert, M. 2008. Real-time shading with filtered importance sampling, Computer Graphics Forum, Vol. 27, No. 4, pp. 1147–1154.

[22] Bagher, M.M., Snyder, J., Nowrouzezahrai, D. 2016. A non-parametric factor microfacet model for isotropic brdfs, ACM Transactions on Graphics, Vol. 35, No. 5, pp. 159:1–159:16.

[23] Bilgili, A., Öztürk, A., Kurt, M. 2011. A general brdf representation based on tensor decomposition, Computer Graphics Forum, Vol. 30, No. 8, pp. 2427–2439.

[24] Jeremias, P., Quilez, I. 2013. Shadertoy: live coding for reactive shaders. ACM SIGGRAPH 2013 Computer Animation Festival, ACM, NY, USA, 1–1.

[25] Parisi, T. 2012. WebGL: Up and Running. 1st, O'Reilly Media, Inc.

[26] Wikipedia, F.I. 2011. WebGL (Web Graphics Library). https://en.wikipedia.org/wiki/WebGL (Access Date: 26.04.2019).

[27] Kajiya, J.T. 1986. The rendering equation, Computer Graphics, Vol. 20, No. 4, pp. 143–150. (Proc. SIGGRAPH '86).

[28] Alekseev, A. 2014. Physically Based Rendering. https://www.shadertoy.com/view/XsfXWX (Access Date: 26.04.2019).

[29] Debevec, P. 1998. Rendering synthetic objects into real scenes: bridging traditional and image-based graphics with global illumination and high dynamic range photography. The 25th Annual Conference on Computer Graphics and Interactive Techniques, ACM, NY, USA, 189–198.