Rendering Optical Effects Based on Spectra Representation in Complex Scenes

Weiming Dong

Rendering Optical Effects Based on Spectra Representation in Complex Scenes

Weiming Dong*

Project ALICE, INRIA Lorraine/Loria, France
Weiming.Dong@loria.fr

Abstract. Rendering the structural color of natural objects or modern industrial products in the 3D environment is not possible with RGB-based graphics platforms and software and very time consuming, even with the most efficient spectra representation based methods previously proposed. Our framework allows computing full spectra light object interactions only when it is needed, i.e. for the part of the scene that requires simulating special spectra sensitive phenomena. Achieving the rendering of complex scenes with both the full spectra and RGB light and object interactions in a ray-tracer costs only some additional fractions of seconds. To prove the efficiency of our framework, we implemented a "Multilayer Film" in a simple ray-tracer. However, the framework is convenient for any complex lighting model, including diffraction or fluorescence.

1 Introduction

Many phenomena and materials, in nature or in industry, have complex optical effects, namely interference, diffraction, fluorescence, dispersion, phosphorescence, etc. These physical effects cannot be rendered with current RGB-based graphics platforms and software. However, as pointed in [1] and [2], color computations in a renderer have to be performed in spectral space if the output is to be used for predictive purpose.

To simulate these optical effects, some researches have been focused on the full spectra representation of light and objects [1]. These methods are not tractable when a complex scene has to be rendered. Recently in [3], for example, a multilayer films model is coded with their efficient spectra representation [4] and implemented in a popular RGB-based renderer. However, the optical effects of the object presented (insect) are computed independently of its 3D environment, although they are highly dependent on the lighting conditions of this environment and interactive with it. Our rendering process overcomes this problem.

Our idea is based on the observation that usually, only a part of a scene needs to be simulated with a full spectra representation. In Figure 1, only a part of the insect has a structural color. In Figure 2, the insects are even partly hidden. In the two figures, color of leaves and flowers, as well as some parts of the insect itself, are due to pigmentation. Moreover, the light transport needs

* This research was done when Weiming Dong was a visiting student in the Sino French Lab in Computer Science, Automation and Applied Mathematics (LIAMA).

H.-P. Seidel, T. Nishita, and Q. Peng (Eds.): CGI 2006, LNCS 4035, pp. 719–726, 2006.

to be simulated with full spectra representation, only when the optical effects of the material have a visual impact on other objects.

This paper presents the rendering process of complex scenes with both the full spectra and RGB light and object interactions. Unlike conventional systems that perform color calculations with tristimulus color values, our system allows to embed any kind of representation of the spectra to calculate colors of complex optical effects. Unlike spectral rendering systems previously proposed [1], our rendering process allows to use and compute spectra in complex scenes only when it is needed, i.e. when they have a significant impact in the transport process, and the final image.

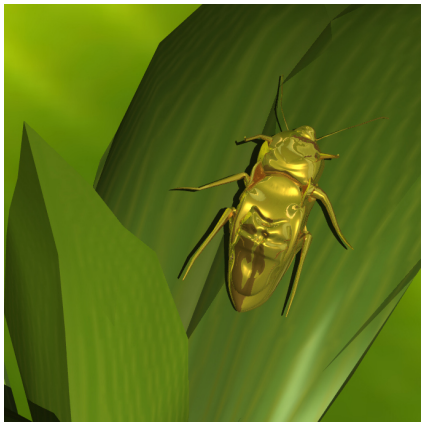

Fig. 1. The *Edgar Poe Gold Bug* is rendered with a full spectral function representation while the main part of the 3D scene is computed with RGB models

(a) (b)

Fig. 2. Two examples of our framework

In the following sections, we first discuss some related works. Then we describe the color representation models used in our system, and also give the whole work flow of the render. We show some images generated by our system and address the advantages comparing with the traditional methods in the fifth section. Finally we draw the conclusion and discuss about the future work.

2 Related Works

Many works have been focused on developing physically based lighting models and perceptually based rendering procedures for computer graphics that will produce synthetic images that are visually and measurably indistinguishable from real-world images [1][5][6][7]. Greenberg et al. [8] proposed a framework which subdivided the whole rendering process into three sub-sections: the local light reflection model, the energy transport simulation, and the visual display algorithms. Glassner provided a mathematical framework for phosphorescence and fluorescence [9][2]. Stam [10] developed a reflection models for metallic surfaces that handle the effects of diffraction. To accurately simulate the optical phenomena, Sun et al. [4] proposed a rendering framework which emphasized real spectra as input, retains full spectral light-object interactions, and generates spectral images (convertible to RGB images for display). This framework is capable of handling wavelength-related optical effects including dispersion, interference, diffraction, and fluorescence, but still very costly, especially when objects with complex spectrum based materials are present only in a part of the scene. Another problem is to render the scene users need to set all the object materials with spectrum, usually it is difficult to get all the spectral data of the materials in the scene.

An accurate and efficient spectral representation is required in our framework. Many methods have been proposed like sampling [11][12], linear model representation [13], and using polynomials [14]. Unfortunately these methods all have difficulties in balancing the accuracy and efficiency. To overcome the drawbacks of the previous methods, Sun et al. proposed a composite spectral model by decomposing all spectra into a smooth background and a list of spikes [15][4][3]. They represented the smooth part with Fourier coefficients and a spike is specified by its location and height. We also use this spectral representation model in our framework.

We use Hirayama et al.'s work [16] for rendering objects coated with multilayer thin films to test our framework. Their method is based on wave optics, and is able to accurately visualize the optical effects of multilayer films.

In [3], a multilayered films is also proposed. The model has several new nice properties. However, the complex optical effects of the *Morpho Rethenor* that is chosen as an illustration cannot be rendered with an approximated interference scheme. In fact, the irregularity in the ridge height of the *Morpho Rethenor* eliminates the interference among the ridges, which results in the diffuse and broad reflection of a uniform color [17]. So due to its complex microstructure, the structural color combines both diffraction and interference in a complex microstructure.

3 Efficient Renderer for Rendering Natural Scenes

We develop a new framework for realistic rendering. In this framework, we use RGB and spectrum together to represent the materials, light sources and the pixel colors of the synthesized image.

3.1 Materials and Light Sources

First, to simulate the natural phenomena which can not be accurately calculated with simple RGB models, and protect the efficiency if the objects also have some RGB oriented properties at the same time, we allow RGB and spectrum to work together to represent one material, different element of the material can have different type of representative model. This means that we set the "natural" part of the material with spectral model and let the other parts still be represented by RGB. For example, considering an object with both interference effect and simple diffuse appearance, we integrate the interference part with the related physical based model (like a spectrum based BRDF model) and define the diffuse value which is independent of wavelength with RGB.

To synthesize realistic images of natural phenomena, we use light sources described with spectral power distributions (SPDs) in the scenes which have spectral effects. Like Sun's spectrally based framework [4], we also recommend the using of real spectral data in order to ensure the accuracy of spectral effect simulation.

At the same time, we also allow RGB light sources in the same scene, this feature is to facilitate the use of monochromatic light sources and the approximation of the real light sources when there is no getatable spectral data. The user can simply choose the RGB color of the light source and convert it into spectrum in the rendering process. In our system, we use Sun's method to derive spectra from colors [15].

3.2 Color Representation

Based on Huygens' principle of independent propagating of light [18], we separately calculate the effect of the light sources, no matter it is an RGB or spectral light source. Then we integrate the effect together as the final results of the local illumination. In our system, we decompose the color of one point into two parts: the RGB part and the spectrum part. Formally the color is the sum of the two parts. We use it as the intermediate format during the rendering process. The color of one point $\boldsymbol{x}$ can be written as follows

$$Color_{inter}(\boldsymbol{x}) = Color_{RGB}(\boldsymbol{x}) + Color_{spectrum}(\boldsymbol{x})$$

where $Color_{RGB}(\boldsymbol{x})$ is the RGB value generated by the interaction between the RGB based material elements and the light sources (represented with RGB model), and $Color_{spectrum}(\boldsymbol{x})$ is the spectral effect caused by the spectral elements of the material.

In fact, for one light source, we first calculate the irradiance of it at the point. Then the radiance (color) of the point will be evaluated according to the type of

the material element. If the element is RGB based, we convert the SPD of the spectral light source into RGB and add the value (radiance) to the RGB part of the point color. On the other hand, the RGB light source should be converted to spectrum if the material element is spectrum based. So the color of the point generated by one light source can be written as

$$Color_{inter}(\boldsymbol{x}) = \sum_{i=1}^{m} Color^{i}_{RGB}(\boldsymbol{x}) + \sum_{j=1}^{n} Color^{j}_{spectrum}(\boldsymbol{x})$$

where m is the number of the RGB based material elements, $Color^{i}_{RGB}(\boldsymbol{x})$ is the RGB radiance (represented with RGB model) generated by the light source, m is the number of the spectral material elements, and $Color^{j}_{spectrum}(\boldsymbol{x})$ is the spectral value which is computed (represented with spectrum) according to the spectral function. So we write the final equation of the color at one point generated by multiple light sources as follows

$$Color_{inter}(\boldsymbol{x}) = \sum_{k=1}^{N}(\sum_{i=1}^{m} Color^{i}_{RGB}(\boldsymbol{x}) + \sum_{j=1}^{n} Color^{j}_{spectrum}(\boldsymbol{x}))$$

where N is the number of light sources in the scene.

3.3 Acceleration

To accelerate the rendering process at run time, we save both the RGB value and the spectral value of all the light sources in the pre-processing step, so we need not do the spectrum-to-RGB or RGB-to-spectrum operation when the irradiance of one point is being calculated. The only work we need to do is to choose the proper value corresponding to the type of the material element.

4 Rendering Pipeline

The whole rendering pipeline comprises three stages: preprocessing, rendering and color transformation. In the first stage, the SPD of the spectral light sources and the spectral functions of the spectral material elements are represented through loading data from the spectral database. The intensity of the RGB light sources and the RGB parts of the materials are also set by the user (or from texture). Then we pre-compute the RGB value of the spectral light sources and the spectra of the RGB light sources. We save these values together with the original data. The second stage is most important: here an intermediate image is generated based on local and global illumination models with ray tracing. The intermediate image is similar to a color image except that for every pixel the information is the combination of a spectrum and a RGB value instead of a color. In the rendering process, when calculating the reflectance intensity, if the reflectance of the material is RGB based, we need to convert the spectral part of the reflectance intensity gathered by the reflected ray into RGB. Contrarily,

if the reflectance of the material is spectrum based, we need to convert the RGB part of the reflectance intensity gathered by the reflected ray into spectrum. The same for the transmittance calculation. Finally we transform the spectral part of the intermediate image into RGB [15] and plus the previous RGB part. This will generate an RGB image for displaying on the screen.

Compared to previous frameworks [8][4], the intermediate color format, the separate light-material element interaction and the intermediate image in our pipeline are new elements. Note that we can also store the spectral part of the intermediate image as a spectral image like [4] to identify errors for particular wavelengths and finding effective improvements. On the other hand, compared with Sun et al.'s [4] framework, we only add a RGB data which can be stored by three "double" variables for each pixel in the rendering process, the memory increase will not be a problem.

Table 1. The information of result images

Items	Figure 1	Figure 2(a)	Figure 2(b)
Triangle Number of Plant	18086	32848	60894
Triangle Number of Insect	49832	49832	49832
Resolution (Pixels)	680×680	600×600	600×600
Our Rendering Time (Seconds)	7.63	12.84	15.97
Sun's Rendering Time (Seconds)	80.23	188.49	234.13
Our Rendering Time without the Insect (Seconds)	3.12	11.02	13.08
Sun's Rendering Time without the Insect (Seconds)	45.66	173.38	198.52

5 Results and Discussion

Figure 1 shows a natural scene with an Edgar Poe Golden Bug rendered by our system, the material is constructed by coating cooper with a 500 nm gold film [16], and the specular value for the high light is an RGB value [1]. The plant and the background are both constructed with RGB models. A parallel light source with the spectral distribution of the CIE standard illumination D_{65} [2] is set above the plant. It will cause wavelength computation only when the traced ray intersects with the bug. Figure 2(a) shows one bug on a plant with many flowers. In this scene, only the the bug is integrated with spectrum based material. The material is the same as the bug in Figure 1, but the position of the camera and the light source (also D_{65}) is different. We can see the different appearance of the iridescence caused by the thin film. Figure 2(b) is another example, we change the thickness of the gold film to 300nm, one can see the appearance is different with the previous two images. We can also notice that in the two images of Figure 2, the insects only occupy a very small part of the scene. Here our system

[1] The original insect model was downloaded from http://www.turbosquid.com. All the plant models used in this paper were downloaded from http://www.toucan.co.jp/indexE.html

is much more efficient than the full spectra rendering framework [4] while the image quality is almost the same. The rendering information of the results is shown in Table 1. All the images are generated on a PC of P4 3.2GHz and 1GB RAM. One can see that our system is nearly 15 times faster than the system of [4]. And we can also see that adding the insect to the scene will only cost a very few additional time comparing with the whole rendering time if the insect will only occupy a small part of the rendering window (for Figure 2(a) 1.82 seconds and for Figure 2(b) 2.89 seconds).

6 Conclusion and Future Work

This paper proposed an efficient framework for realistic image synthesis which can use real spectral data and RGB value together as input, retains full spectral interactions between lights and the spectral parts of the material of the objects, and generate images described with a format combining of both RGB and spectrum. We have shown that this framework suffices to describe the natural optical effects in realistic image synthesis, and have facilitated its practical application through a new color representation model - the combination model. Unifying previous research on traditional and spectral modeling and rendering, this new framework provides a useful and efficient basis for simulating general complicated phenomena.

A lot of work is needed to demonstrate the efficiency of the method and to control the visual impact of the rendering in complex global illuminated environments. We plan to implement and test our rendering process in a Photon Mapping Renderer [19] in complex geometric and physical scenes. Our tests use the the *Thin Film Model* for iridescence [16] and the Sun et al.'s dual method for spectra functions coding [4]. Improvements can be done in theses two areas. In particular, design a complex biology based micro structure model that can combines interference and diffraction is still an open problem as well as in Computer Graphics than in Optical Engineering.

References

1. Devlin, K., Chalmers, A., Wilkie, A., Purgathofer, W.: State of the art report: Tone reproduction and physically based spectral rendering. In: Proceedings of Eurographics 2002. (2002) 101–123
2. Glassner, A.S.: Principles of Digital Image Synthesis. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA (1995)
3. Sun, Y.: Rendering biological iridescences with rgb-based renderers. ACM Trans. Graph. (2006)
4. Sun, Y., Fracchia, F.D., Drew, M.S., Calvert, T.W.: A spectrally based framework for realistic image synthesis. The Visual Computer **17**(7) (2001) 429–444
5. Hall, R.A., Greenberg, D.P.: A testbed for realistic image synthesis. **3**(8) (1983) 10–20
6. Rougeron, G., Péroche, B.: An adaptive representation of spectral data for reflectance computations. In: Proceedings of the Eurographics Workshop on Rendering Techniques '97, London, UK, Springer-Verlag (1997) 127–138

7. Iehl, J.C., Péroche, B.: An adaptive spectral rendering with a perceptual control. Comput. Graph. Forum **19**(3) (2000)
8. Greenberg, D.P., Torrance, K.E., Shirley, P., Arvo, J., Lafortune, E., Ferwerda, J.A., Walter, B., Trumbore, B., Pattanaik, S., Foo, S.C.: A framework for realistic image synthesis. In: SIGGRAPH '97: Proceedings of the 24th annual conference on Computer graphics and interactive techniques, New York, NY, USA, ACM Press/Addison-Wesley Publishing Co. (1997) 477–494
9. Glassner, A.S.: A model of fluorescence and phosphorescence. In: Proceedings of the 5th Eurographics Workshop on Rendering., Berlin Heidelberg New York, Springer (1994) 57–68
10. Stam, J.: Diffraction shaders. In: SIGGRAPH '99: Proceedings of the 26th annual conference on Computer graphics and interactive techniques, New York, NY, USA, ACM Press/Addison-Wesley Publishing Co. (1999) 101–110
11. Cook, R.L., Torrance, K.E.: A reflectance model for computer graphics. ACM Trans. Graph. **1**(1) (1982) 7–24
12. Meyer, G.W.: Wavelength selection for synthetic image generation. Comput. Vision Graph. Image Process. **41**(1) (1988) 57–79
13. Peercy, M.S.: Linear color representations for full speed spectral rendering. In: SIGGRAPH '93: Proceedings of the 20th annual conference on Computer graphics and interactive techniques, New York, NY, USA, ACM Press (1993) 191–198
14. Raso, M.G., Fournier, A.: A piecewise polynomial approach to shading using spectral distributions. In: Graphics Interface 91, Toronto, Canada, Canadian Information Processing Society (1991) 40–46
15. Sun, Y., Fracchia, F.D., Calvert, T.W., Drew, M.S.: Deriving spectra from colors and rendering light interference. IEEE Comput. Graph. Appl. **19**(4) (1999) 61–67
16. Hirayama, H., Kaneda, K., Yamashita, H., Yamaji, Y., Monden, Y.: Visualization of optical phenomena caused by multilayer films with complex refractive indices. In: PG '99: Proceedings of the 7th Pacific Conference on Computer Graphics and Applications, Washington, DC, USA, IEEE Computer Society (1999) 128–137
17. Kinoshita, S., Yoshioka, S., Kawagoe, K.: Mechanisms of structural colour in the morpho butterfly: cooperation of regularity and irregularity in an iridescent scale. Proc. R. Soc. Lond. B 266 **269**(1499) (2002) 1417–1421
18. Baker, B.B., Copson, E.T.: The Mathematical Theory of Huygens' Principle. Second edn. Oxford University Press, Oxford, England (1950)
19. Jensen, H.W.: Realistic Image Synthesis Using Photon Mapping. A. K. Peters, Ltd., Natick, MA, USA (2001)