ISSN 0361-7688, Programming and Computer Software, 2011, Vol. 37, No. 5, pp. 236–244. © Pleiades Publishing, Ltd., 2011.
Original Russian Text © D.D. Zhdanov, I.S. Potemin, V.A. Galaktionov, B.Kh. Barladyan, K.A. Vostryakov, L.Z. Shapiro, 2011, published in Programmirovanie, 2011, Vol. 37, No. 5.

Spectral Ray Tracing in Problems of Photorealistic Imagery Construction

D. D. Zhdanov[a], I. S. Potemin[b], V. A. Galaktionov[b], B. Kh. Barladyan[b], K. A. Vostryakov[b], and L. Z. Shapiro[b]

[a] *Vavilov State Optical Institute, Birzhevaya liniya 12, St. Petersburg, 199034 Russia*
[b] *Keldysh Institute of Applied Mathematics, Russian Academy of Sciences, Miusskaya pl. 4, Moscow, 125047 Russia*
e-mail: ddzhdanov@mail.ru, ipotemin@yandex.ru, vlgal@gin.keldysh.ru, k.vostryakov@gmail.com

Received March 3, 2011

Abstract—This paper analyzes color rendering problems arising in computer imaging of scenes with complex optical properties. The sources of these errors for imagery constructed on the basis of the RGB space are analyzed and an efficient and correct algorithm is proposed for spectral imagery. These solutions are applied to problems of direct stochastic and inverse deterministic ray tracing. In addition, complex scenes with mixed optical properties defined in different (including RGB) models are proposed to be treated by an efficient and correct algorithm for reducing all optical properties into a single spectral model.

DOI: 10.1134/S0361768811050069

1. INTRODUCTION

Construction of photorealistic imagery of complex scenes containing objects with special properties of transmission, reflection, and scattering requires physically correct models for calculating the luminance formed by given objects. Normally, computer programs for imaging use an RGB representation for describing optical properties of the scene and sources of light. This approximation is used not only in "hardware" systems based on OpenGL, but also in most programs claiming that the emerging images are physically correct and photorealistic [2, 3].

It is known that the RGB space is bounded, and the representation of positive colors does not cover the entire visible spectral range. In this context, it is necessary to distinguish between two factors related to clipping and distortion of the color representation.

The first factor is color distortion arising in the calculation of visible luminance of scene objects. The luminance in the RGB representation is proportional to the product of R, G, and B components of object illumination on the corresponding components of the object luminance factor in the observation direction. This relation does not lead to color distortion only if the luminance factor was originally defined for the color of the given source of light. Clearly, for scenes observed through color glasses or mirrors or for scenes containing radiation sources of different colors, the color distortions cannot be excluded [4].

The second factor is the color clipping on radiation receivers. The majority of modern receivers cannot reproduce visible luminance or illumination without color distortion. Such radiation receivers as CCD matrices transform spectral distribution of the illumination into an RGB image. Naturally, this conversion leads to the loss of the negative color component, which cannot be corrected. Therefore, when speaking about formation of photorealistic imagery, we should realize that the "photorealisticness" may have already involved color distortion, and the construction of photorealistic images should incorporate a suitable model of radiation receiver [5].

Evidently, the construction of photorealistic images incorporates two independent models. These are a model of construction of a true distribution of illumination or luminance on a radiation receiver and a model of transformation of the input radiation into an RGB-representation. This study is devoted exclusively to the first aspect, namely, to the formation of the distribution of illumination and luminance without color distortion. The only model ensuring that the color rendering is correct is the spectral model, and the only method for calculating the illumination and luminance is the spectral ray tracing. Thus, the main focus of this paper is on the physically correct tracing of spectral rays in scenes containing objects with complex optical properties.

2. BASIC LIMITATIONS OF THE RGB MODEL AND ERRORS IN COLOR CALCULATION

The predominant majority of computer systems for image construction use an additive RGB representation of color in terms of optical properties of scene objects. Scene objects with color properties can be divided into two groups. The first group includes primary sources of light radiation, and the second group

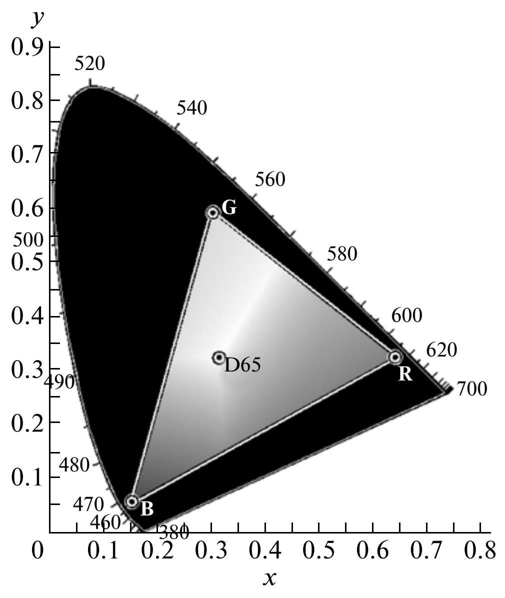

Fig. 1. The region of positive RGB colors in the visible spectral range.

includes secondary sources of light reflection, refraction, and scattering.

In the case of a primary source of light, the RGB color is calculated by using the well-known ICE formulas [6]

$$\left.\begin{aligned} x &= \int_{380}^{780} \bar{x}_\lambda \varphi(\lambda) d\lambda \\ y &= \int_{380}^{780} \bar{y}_\lambda \varphi(\lambda) d\lambda \\ z &= \int_{380}^{780} \bar{z}_\lambda \varphi(\lambda) d\lambda \end{aligned}\right\} \Rightarrow rgb = M \cdot xyz, \qquad (1)$$

where $\bar{x}_\lambda$, $\bar{y}_\lambda$, and $\bar{z}_\lambda$ are the sensitivity functions *XYZ* for the standard observer; $\varphi(\lambda)$ is the density of distribution of luminance (or other photometric parameter); *x*, *y*, and *z* are color coordinates in the *XYZ* space; and *M* is the matrix of *XYZ*-to-RGB transformation of color coordinates [12].

It is known that, although color representation as RGB channels covers the entire visible spectral range, the color components can be negative numbers. Figure 1 demonstrates the so-called color triangle indicating coordinates of all positive R, G, and B colors in the (*x*, *y*)-chromatic system of coordinates as well as the spectral area containing negative values of R, G, or B channels.

Like in digital image mapping systems, computer simulations use positive values for mapping an RGB color. It is clear that this limitation of the color representation leads to errors in the color mapping. The color-sensation errors are normally estimated using the color expression in the Lab space. The International Commission on Illumination (CIE, Commission Internationale de l'Eclairage) defined a series of color-difference standards: CIE76 [7], CIE94, and CIEDE2000 [8]. Using the color-difference formulas, one can estimate the color-sensation error (ΔE^*_{ab}) caused by clipping of negative values in the RGB space.

In the case of a secondary source of light, it is more correct to speak about the color of surface as the response of surface to incident light. Since surface has no color of its own, the color of surface in the RGB space is normally meant as a normalized color of the reflected (refracted or scattered) light from a standard white source of light (for example, D65). The general algorithm for calculating the RGB color of surface looks as follows:

$$\left|\begin{aligned} x_{D65} &= \int_{380}^{780} \bar{x}_\lambda D65(\lambda) d\lambda \\ y_{D65} &= \int_{380}^{780} \bar{y}_\lambda D65(\lambda) d\lambda \\ z_{D65} &= \int_{380}^{780} \bar{z}_\lambda D65(\lambda) d\lambda \end{aligned}\right\| \begin{aligned} x_\rho &= \int_{380}^{780} \bar{x}_\lambda \rho(\lambda) D65(\lambda) d\lambda \\ y_\rho &= \int_{380}^{780} \bar{y}_\lambda \rho(\lambda) D65(\lambda) d\lambda \\ z_\rho &= \int_{380}^{780} \bar{z}_\lambda \rho(\lambda) D65(\lambda) d\lambda \end{aligned}\right|$$

$$rgb_{D65} = M \cdot xyz_{D65} \quad rgb_\rho = M \cdot xyz_\rho \qquad (2)$$

$$r = \frac{r_\rho}{r_{D65}}; \quad g = \frac{g_\rho}{g_{D65}}; \quad b = \frac{b_\rho}{b_{D65}};$$

where $\rho(\lambda)$ describes spectral properties of the reflection (transmission) surface.

In real conditions, one normally considers complex illumination with sources of light of different colors; in addition, the mirror and refracted color objects add virtual color sources of light. As a result, the actual scene has no source of light the color of which could be used as a base for calculating the colors of scene objects. Therefore, this calculation is normally performed with white color as the base (for example, with the help of source D65). Clearly, this solution may lead to distortion of the color of the object in the actual scene.

3. COLOR RENDERING PROBLEMS IN SOLVING THE RENDERING EQUATION IN THE RGB SPACE

In the previous section, we considered basic problems of color rendering for computer simulations in the RGB space. In this section, we discuss problems related to solving the rendering equation in the RGB space. This equation determines the luminance in the

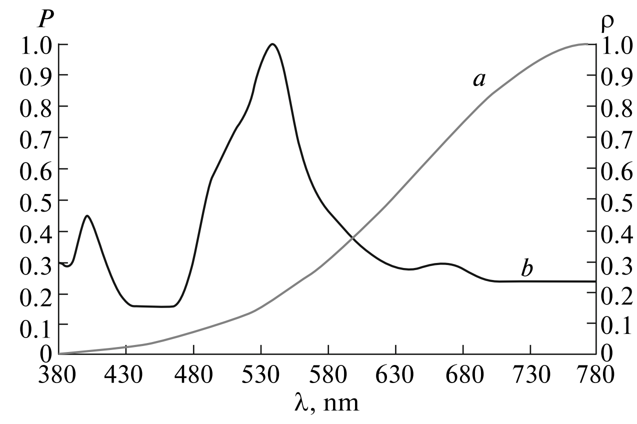

Fig. 2. (*a*) Spectrogram of source of light; (*b*) spectral coefficient of surface reflection.

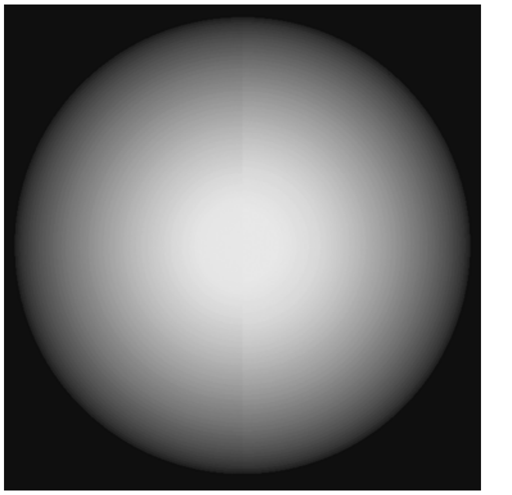

Fig. 3. Image of sphere. The left half is in the RGB space and the right half is in the spectral space.

observation direction as a sum of observed object's own luminance and the luminance scattered by the given object in the observation direction [3, 9]. For static scenes, the rendering equation governing the luminance of the object at the point $\vec{p}$, in the direction $\vec{v}$, and for the color component c can be written as

$$L(\vec{p}, \vec{v}, c) = \tau(\vec{p}, \vec{v}, c) \times \frac{n_O^2}{n_S^2}\left(L_0(\vec{p}, \vec{v}, c) + \frac{1}{\pi}\int_{4\pi} BSDF(\vec{p}, \vec{v}, \vec{v}', c)E_\omega(\vec{p}, \vec{v}', c)d\omega\right), \quad (3)$$

where $L_0(\vec{p}, \vec{v}, c)$ is observed object's own luminance at the observation point; $\tau(\vec{p}, \vec{v}, c)$ is the transmission (transparency) of the medium between the observer and the point of observation; $BSDF(\vec{p}, \vec{v}, \vec{v}', c)$ is the function of bidirectional scattering from the source of illumination in the direction $\vec{v}'$ to the observer; $E_\omega(\vec{p}, \vec{v}', c)$ is object's local illumination at the observation point in the direction $\vec{v}'$ created by the source of light in the solid angle $d\omega$; and n_O and n_S are refraction indices of the media of the observer and observed surface, respectively.

The main and adequate method of solving the rendering equation is that of ray tracing. In this approach, the integration over the sphere is performed tracing rays in the direction of all possible sources of illumination, and the rendering equation takes the following form:

$$L(\vec{p}, \vec{v}, c) = \tau(\vec{p}, \vec{v}, c) \times \frac{n_0^2}{n_S^2}\left(L_0(\vec{p}, \vec{v}, c) + \frac{1}{\pi}\sum_{for\ all\ rays} BSDF(\vec{p}, \vec{v}, \vec{v}', c)E_i(\vec{p}, \vec{v}', c)\right), \quad (4)$$

where $E_i(\vec{p}, \vec{v}', c)$ is object's local illumination at the observation point in the direction $\vec{v}'$ created by the ith ray.

In the case of modeling in the RGB space, the surface colors ($BSDF(\vec{p}, \vec{v}, \vec{v}', RGB)$ and $\tau(\vec{p}, \vec{v}, RGB)$) are reduced to the same illumination conditions (normally, to the white source D65). This simplification leads to a color reproduction error caused by ambiguous dependence of the spectral composition of radiation on the RGB color. On the example of surface-scattered light luminance, this dependence has the following form:

$$BSDF(\vec{p}, \vec{v}, \vec{v}', RGB)E_i(\vec{p}, \vec{v}', RGB) \neq RGB[BSDF(\vec{p}, \vec{v}, \vec{v}', \lambda)E_i(\vec{p}, \vec{v}', \lambda)]. \quad (5)$$

This means that the use of the RGB space for simulation does not secure accurate color rendering. The following two examples visually show the color distortion caused by calculations in the RGB space. These simple examples make it possible to compare the results of rendering in the RGB space with the results of spectral rendering.

In the first example, the red source of light given in the visible spectral range (the spectrogram of the source of light is shown in Fig. 2a) illuminates a spherical green surface (the spectral coefficient of reflection is shown in Fig. 2b). The RGB color of the green surface was calculated under the condition that the surface is illuminated by the white source D65.

The image of the sphere illuminated by a red source of light is shown in Fig. 3. Because the image is mirror symmetric, for convenience of comparison, the left half of the image was calculated in the RGB space and the right half was calculated in the spectral space.

In this example, the difference in colors arises from the fact that the color of the source of light in the scene differs from the reference color used to calculate the

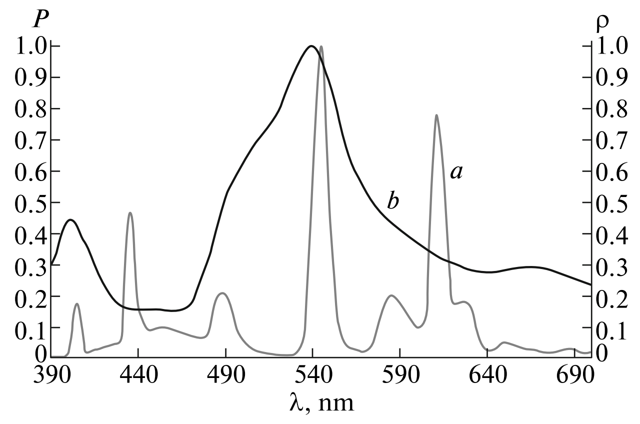

Fig. 4. (*a*) Spectrogram of source of light; (*b*) spectral coefficient of surface reflection.

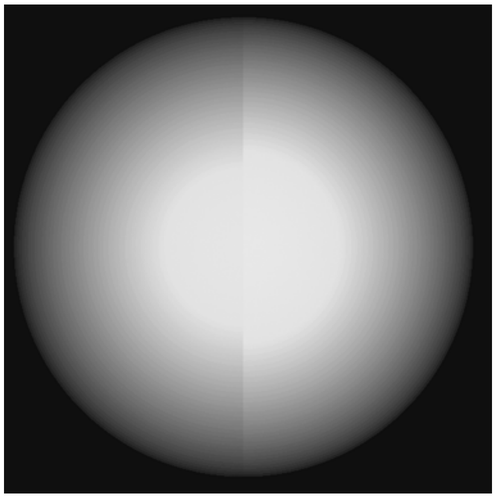

Fig. 5. Image of sphere. The left half is in the RGB space and the right half is in the spectral space.

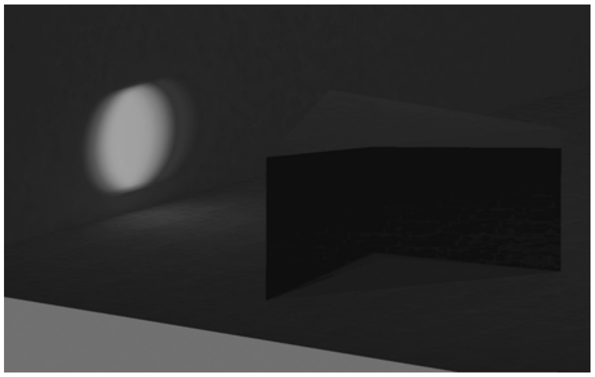

Fig. 6. Image of dispersive spot created by a glass prism. The simulation was conducted in the visible range at 81 wavelengths.

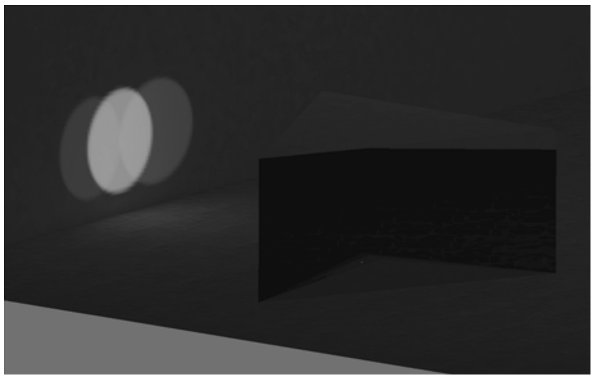

Fig. 7. Image of dispersive spot created by a glass prism. The simulation was conducted in the RGB space.

color of surface. However, the difference in colors may arise even if the source of light is the same. In the next example, the white light (the spectrogram of the source of light is shown in Fig. 4a) illuminates the green spherical surface (the spectral coefficient of reflection is shown in Fig. 4b).

Since the spectrogram of the white color differs from that of the base source D65 (although their RGBs are almost the same), the resulting colors in the RGB and spectral models differ from one another. Similar to Fig. 3, the left half of the image in Fig. 5 was calculated in the RGB space, and the right half was calculated in the spectral space.

The solution of the rendering equation in the RGB space has a key limitation: the dispersion effects (i.e., the dependence of the light propagation direction on the light wavelength) cannot be simulated. In the case of dispersion scattering, the luminance in this direction is equal to zero because the scattering proceeds in an infinitely narrow spectral interval. Therefore, in the case of dispersion, it makes sense to speak about the coefficient of transmission (reflection) for a given wavelength $\tau(\vec{p}, \vec{v}, \lambda)$ (i.e., the ratio of flux density of transmitted (reflected) radiation to flux density of incident radiation).

The construction of scene images containing dispersing elements can be physically correct if the spectral area during the simulation is divided into a large number of narrow spectral areas. Figure 6 shows an image of dispersive spot created by a glass prism illuminated by the white conic source D65.

A similar approach might be applied also to the RGB space, for example, each color channel might be matched by its wavelength (R $\approx$ 620 nm, G $\approx$ 540 nm, B $\approx$ 460 nm). However, the small number of color channels (R, G, and B) leads to discreteness of images, which makes them unnatural. Figure 7 shows

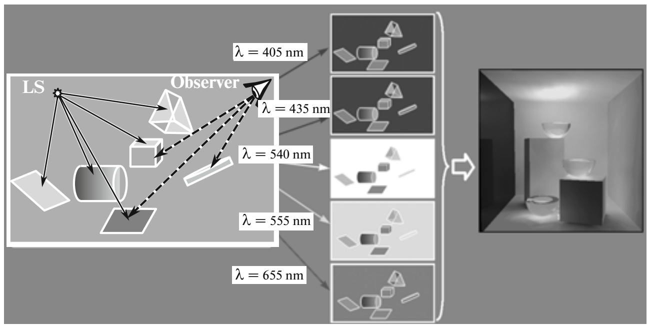

Fig. 8. Schematic of the first technique for solving the equation of rendering in the spectral space.

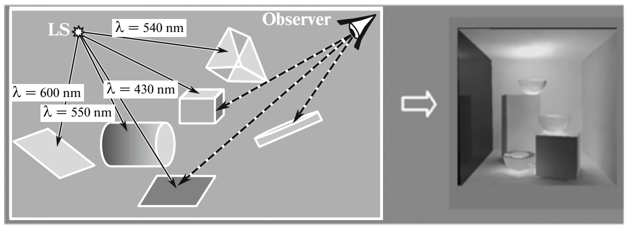

Fig. 9. Schematic of the second technique for solving the equation of rendering in the spectral space.

a similar model of dispersive scattering of a glass prism accomplished in the RGB space.

4. SOLVING THE RENDERING EQUATION IN THE SPECTRAL SPACE

A physically correct solution of the rendering equation can be found only in the spectral space. Technically, the rendering equation (3) can be solved in three ways. First, the ray tracing technique can be used for a number of "monochromatic" scenes, the optical properties of which are defined for one wavelength. The final result is the sum of individual "monochromatic" calculations. Second, for a scene the optical properties of which are defined for a wider spectral range, individual "monochromatic" rays (i.e., rays whose properties are defined for one wavelength) can be traced. Of course, the "monochromatic" rays should cover the entire spectral interval and contribute a suitable portion of the spectral energy to the final result. Third, for a scene the optical properties of which are defined for a wider spectral range, "polychromatic" rays (i.e., rays whose spectral properties are defined in the entire spectral interval) can be traced. In this case, the final result is just the result of "polychromatic" calculations.

Let us consider in detail the pros and cons of these techniques. Schematically, the first solution (see Fig. 8) seems to be the simplest one.

To begin with, N independent scenes with optical properties corresponding to given wavelengths are formed, where N is the number of wavelengths used in calculations. Optical properties of light sources (LSs) are specified as a spectral distribution of radiation flux density. This specification is the most convenient for any type of spectral modeling because the spectral width is excluded and the spectral data can be applied to any set of wavelengths. This modeling results in the formation of N "monochromatic" images that can be combined into an image containing spectral density of radiation for a given set of wavelengths. Using expression (1), an image in the RGB space is formed, which has no color transformation errors in the course of ray tracing. This "monochromatic" solution was implemented as an extension of the SPECTER software [10], which has operated until now exclusively in the RGB space. To generate spectral scenes automatically, we developed a scene generator transforming a set of spectral data into a corresponding set of independent scenes. Then, individual images merged into a single RGB image.

The efficiency of this solution turned out to be insufficient; therefore, the only benefit was the simplicity of its implementation (this technique can be applied to almost any renderer). The next possible solution of the problem of spectral modeling is the tracing of "monochromatic" rays in a spectral scene. The schematic of this solution is shown in Fig. 9.

The "monochromatic" rays are emitted for the whole set of wavelengths defined in the scene; however, unlike in the previous technique of spectral modeling, the energy of rays is determined by the spectral flux rather than by the spectral flux density. The need in denormalization from flux density to the very flux value is explained by the fact that the interval between neighboring wavelengths specified in the scene may vary. Expression (6) shows a simple way of denormalization of the spectral flux density:

$$P(\lambda_i) = \varphi(\lambda_i)\frac{\lambda_{i+1} - \lambda_{i-1}}{2}, \qquad (6)$$

where $\varphi(\lambda_i)$ is the spectral density of flux $P(\lambda_i)$.

When the calculations are terminated, the flux transmitted by the ray is normalized back, which makes it possible to obtain the result as the spectral density of a given light value and use expression (1) for representing the result in the RGB space. Depending on the spectral modeling technique (deterministic or stochastic, direct or inverse), the ray can transmit both spectral energy and unit energy, with the probability of its emission being proportional to energy in the spectral interval.

Clearly, the only advantage of this solution is its operational convenience. All spectral data are encapsulated in a single scene, and the image is constructed in the course of a single spectral calculation. However, the efficiency of spectral calculations does not exceed the efficiency of the previous solution.

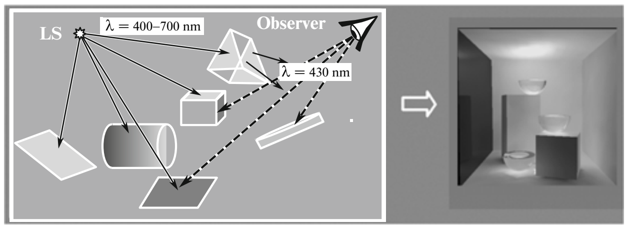

Fig. 10. Schematic of the third technique for solving the equation of rendering in the spectral space.

In this work, we propose an efficient solution for spectral ray tracing in scenes defined in the entire spectral range. This solution implies tracing of a "polychromatic" ray. The schematic of the ray-tracing algorithm is presented in Fig. 10.

The idea of the method is as follows. A source of light or a camera emits a ray defined in the entire spectral interval. This ray propagates in the scene and, on all nondispersive surfaces, is transformed for the entire spectral range. This transformation is similar to the color transformation in the RGB space when the R, G, and B components are replaced by the normalized spectral density. In the case of deterministic ray transformation (for example, calculation of visible luminance), this is a simple product of denormalized spectral light characteristics of the ray by the corresponding values of spectral coefficients of the surface.

In the case of stochastic direct ray tracing, the situation is more complicated. First, in accordance with spectral powers K of the light sources, a source i is chosen, which emits a ray:

$$\frac{\sum_{j=1}^{i-1}\sum_{\lambda}P_j(\lambda)}{\sum_{j=0}^{K}\sum_{\lambda}P_j(\lambda)} \le \xi < \frac{\sum_{j=1}^{i}\sum_{\lambda}P_j(\lambda)}{\sum_{j=0}^{K}\sum_{\lambda}P_j(\lambda)}, \tag{7}$$

where ξ is a random number uniformly distributed between 0 and 1.

Similarly, the starting point, direction, and spectral composition of the ray are selected in a probabilistic way. Then, the spectral energy of the ray is normalized with respect to the unit energy:

$$p(\lambda_n) = \frac{P_i(\lambda_n)}{\sum_{l=1}^{N}P_i(\lambda_l)}. \tag{8}$$

When there is no dispersion, the ray retains the entire spectral range of wavelengths for any transformations on the object. In this case, the spectral transformation is similar to the ray transformation in the RGB space. This means that, in accordance with the spectral characteristics of the scattering object and ray spectrogram, the direction of the new ray and its spectral composition are chosen. The spectral power of the ray is normalized to the unit power (8). The schematic of the process of spectral scattering on a diffusive surface is shown in Fig. 11a.

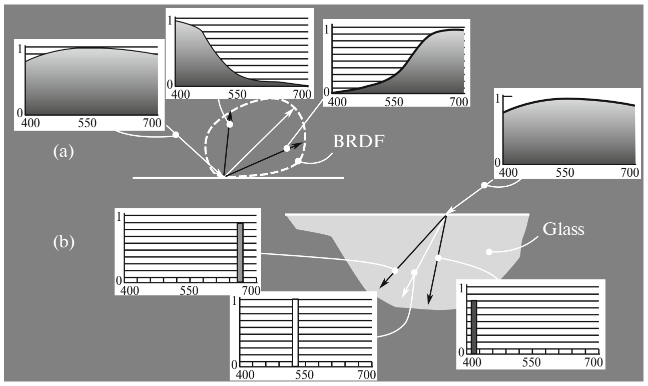

Fig. 11. Two ways of transformation of spectral rays on the surface: (a) the case of diffusive scattering and (b) the case of dispersive scattering.

The case of spectral transformation of the ray on a dispersing surface is more interesting. For these surfaces, there is no scattering direction covering the whole spectral composition of the ray. Consequently, the only solution is to transform "polychromatic" rays into "monochromatic" ones. The schematic of this process is shown in Fig. 11b. In the case of deterministic tracing, at the point of dispersion, a cluster of "monochromatic" rays is generated, the number of which is equal to the number of wavelengths in the spectral ray composition. Each ray has its own direction and is recursively traced from this point. When the recursive tracing of the final ray is terminated, at the dispersion point, the complete spectral composition of the ray is collected.

In the case of stochastic tracing, the dispersion is processed in a probabilistic way. When the "Russian roulette" algorithm is used, only one ray is traced and no recursion is permitted. Therefore, the "polychromatic" ray representing a wide spectrum is used to extract (with a probability proportional to the denormalized ray energy at wavelengths) a single nth "monochromatic" ray continuing the stochastic propagation in the scene:

$$\sum_{l=1}^{n-1}p(\lambda_l) \le \xi < \sum_{l=1}^{n}p(\lambda_l). \tag{9}$$

When the "Russian roulette" algorithm is used, the energy of the ray (both "polychromatic" and "monochromatic") is normalized to unity. Consequently, the energy transmitted by the unit ray has no reliable information about the spectral composition of radiation. The true spectral composition is formed on radiation receivers as the probability of selection of a ray of a certain wavelength. The spectral characteristics are

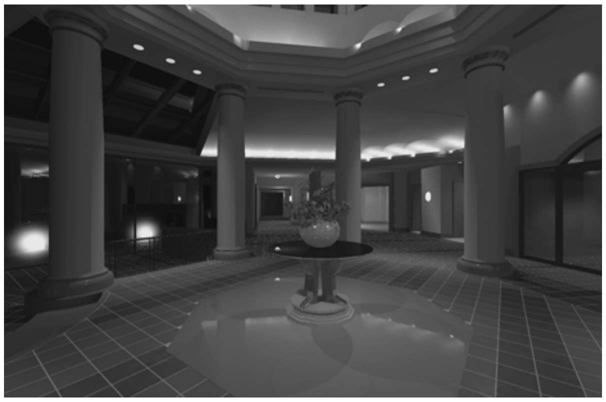

Fig. 12. Image of a scene with spectral properties with no dispersion.

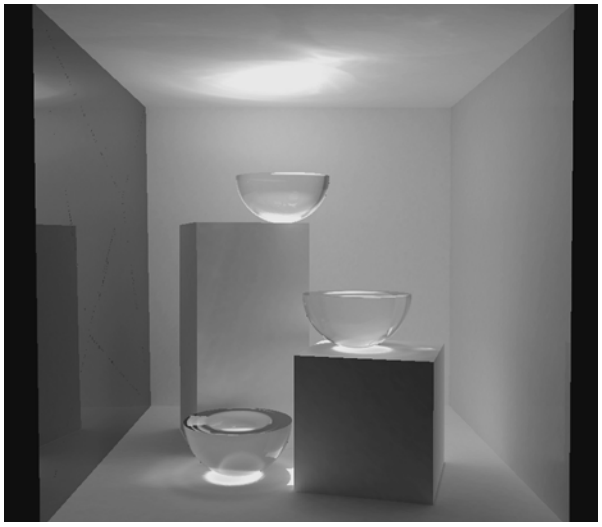

Fig. 13. Image of a scene with dispersive optical elements.

normalized with respect to density (6) of these characteristics on radiation receivers.

The efficiency of this solution for scenes with complex geometry is apparent. In complex scenes, the major part of time (up to 90%) is spent on ray tracing (namely, on the search of the point of ray intersection with geometric objects of the scene). Therefore, although the transformations in the wide spectral range significantly reduce the efficiency of rendering, the total efficiency remains substantially higher than that of rendering based on independent tracing of rays of one wavelength. The dispersion effects dividing the ray defined in the wide spectral range into individual "monochromatic" rays, as a rule, slightly reduce the total efficiency of the program operation. First, for the majority of scenes, the fraction of dispersive elements is insignificant in the total scene volume. Second, the probability of dispersion immediately on the first element of the ray trace is not high.

This solution was implemented in the INSPIRER2 software for the modes of direct stochastic and reverse deterministic ray tracings. This made it possible to fulfill physically correct and efficient rendering of spectral scenes with account of direct illumination.

In the framework of this study, a method of spectral modeling based on tracing a "polychromatic" ray in a spectral scene (extension of INSPIRER2) was implemented. This method was compared with the method based on the calculation of a set of independent "monochromatic" scenes (used in the SPECTER software).

The images generated by these methods are indistinguishable; however, their computational performances are different. Below, two typical images of spectral scenes are shown. Figure 12 demonstrates a scene with no dispersion effects, and Fig. 13 shows a scene with dispersion elements (normally, the dispersion elements occupy the lesser part of the scene volume). The spectral calculations were performed on 41 wavelengths.

The table presents values of rendering times (for both direct and back tracing of rays) obtained on an Intel Core2 Duo T9400 2.53 GHz computer.

It can be seen that the solution used for "polychromatic" ray tracing is much more efficient than the calculations performed at individual wavelengths. In addition, the slowdown caused by the dispersion effect slightly affects the total time of rendering.

Time of rendering for different types of scenes and models of spectral calculations

Model of spectral calculation	Scene, Figure 12	Scene, Figure 13
"Monochromatic" tracing of individual scenes	52.3 min	19.4 min
"Polychromatic" ray tracing in combined spectral scene	16.9 min	6.8 min

5. SPECTRAL RAY INTERACTION WITH RGB OBJECTS OF THE SCENE

In the majority of cases of dealing with scenes containing complex environmental objects (such as ordinary or HDRI textures), the optical properties of the scene in the spectral representation cannot be fully determined. Here, we arrive at the question of an optimal solution for correct rendering (although complete physical correctness is impossible).

There exist two basic solutions. The first solution is simultaneous rendering in the two (spectral and RGB) spaces. That is, all that can be traced in the spectral space is traced in this space. The remainder is traced in the RGB space. This results in two—spectral and RGB—images. The final image in the RGB space is the sum of two images. The second solution converts

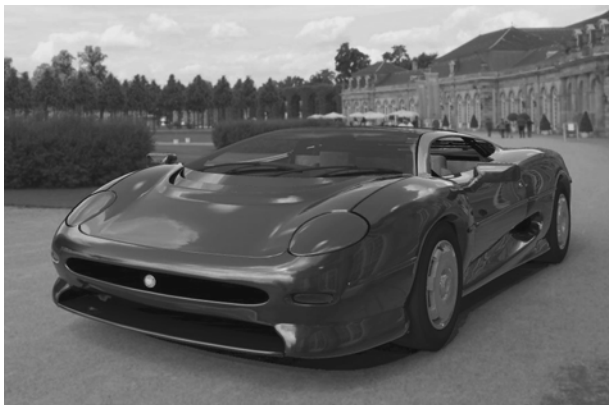

Fig. 14. Example of rendering performed in a space with mixed RGB and spectral data.

all optical properties determined in the RGB space into a spectral representation and fulfils ray tracing exclusively in the homogeneous spectral space.

In this problem, our choice fell on the second solution. The main reasons are as follows: first, it is technically simpler, and, second, it is slightly more accurate because avoids intermittent color clipping, which may arise in turning from the spectral to RGB space.

Apparently, the inverse (RGB-to-spectrum) conversion (1) is ambiguous, and there exist an infinite number of spectral curves that can generate one and the same RGB color. Therefore, the main criteria of the RGB-to-spectrum conversion are maximum uniformity of spectral data and, for optical properties of surfaces, keeping the coefficient of spectral reflection (transmission) below (or equal to) one.

Many solutions for the RGB-to-spectrum conversion have been proposed. For example, the author of [11] proposed to generate a spectral curve as the sum of spectral curves of the major colors (red, green, blue, yellow, cyan, and magenta). Although the resulting spectral curve is sufficiently uniform, the shortcoming of this and a number of other methods is that the surface color is transformed inaccurately, which is caused by the fact that the coefficient of reflection (transmission) takes its maximum permissible value.

In this work, we proposed an efficient and correct solution making it possible to generate spectral curves from RGB data. Since the spectrum-to-RGB conversion is linear, the spectral distribution can be expressed as

$$P(\lambda_l) = w_l^R \cdot r + w_l^G \cdot g + w_l^B \cdot b, \qquad (10)$$

where coefficients w_l^R, w_l^G, and w_l^B are determined from the conditions of minimum variation in the spectral curve and reversibility.

The RGB-to-spectrum conversion is performed by two main methods. For simple data (like the coefficients of reflection or transmission), the conversion is performed in the course of scene loading. Therefore, the rendering operates exclusively with spectral data. For complex objects (like texture or HDRI source of radiation), the conversion into the spectral representation is postponed to the time of ray tracing. Therefore, when the ray hits such an object, the RGB data are converted into the spectral representation. Given that expression (10) is not laborious, there is almost no slowdown. In addition, the postponed conversion of color makes it possible to substantially reduce the demand for required random-access memory, which is important for scenes containing high-quality textures or HDRI sources.

As an example of rendering performed in a space with mixed RGB and spectral data, Fig. 14 shows an image of a car calculated in the spectral space at 41 wavelengths. For the paint of the car, a bidirectional spectral reflection function was determined. The car was illuminated by a light source determined as HDRI in the RGB space.

6. CONCLUSIONS

In the majority of cases, the distortion in color rendering makes it impossible to use the RGB space for construction of photorealistic images. Therefore, the only possible solution for physically correct rendering is the spectral ray tracing.

Using the INSPIRER2 software, we implemented an efficient and physically correct algorithm of spectral tracing of backward deterministic and direct stochastic rays, which made it possible to perform spectral rendering in scenes with multiple diffusive scattering and dispersion. In addition, we implemented algorithms of efficient and correct conversion of optical properties of the scene defined in the RGB space into the spectral representation. This allowed us to perform spectral rendering in scenes with complex environment (for example, illuminated by an HDRI source of light).

ACKNOWLEDGMENTS

This study was supported by RF Presidential Grant from the Program for the Support of Leading Scientific Schools (project no. SSh-8129.2010.9), by the Russian Foundation for Basic Research (project no. 10-01-00302), and and by Integra Inc.

The authors are grateful to V.G. Sokolov and A.A. Garbul for providing us with examples.

REFERENCES

1. Kilgard, M.J. and Akeley, K., Modern OpenGL: Its Design and Evolution, *Proc. of SIGGRAPH Asia'08*, 2008.
2. Jensen, H.W. and Christensen, P., High Quality Rendering Using Ray Tracing and Photon Mapping, *Proc. of SIGGRAPH'07,* 2007.

3. Pharr, M. and Humphreys, G., *Physically Based Rendering—From Theory to Implementation*, Morgan Kaufmann, 2004.
4. Reinhard, E., Ward, G., and Johnson, G., Color Imaging, *Proc. of SIGGRAPH'09*, 2009.
5. Reinhard, E., Stark, M., Shirley, P., and Ferwerda, J., Photographic Tone Reproduction for Digital Images, *Proc. of SIGGRAPH 2002*, 2002.
6. Volosov, D.S. and Tsivkin, M.V., *Teoriya i raschet svetoopticheskikh sistem* (Theory and Calculation of Light-Optical Systems), Moscow: Iskusstvo, 1960.
7. Sharma, G., *Digital Color Imaging Handbook*, CRC, 2003.
8. Sharma, G., Wu, W., and Dalal, E.N., The CIEDE2000 Color-Difference Formula: Implementation Notes, Supplementary Test Data, and Mathematical Observations, ECE Dept., Univ. of Rochester, Rochester, NY 14627-0126, USA.
9. Kajiya, J.T., The Rendering Equation, *Proc. of SIGGRAPH 1986*, 1986, p. 143.
10. http://www.integra.jp/en/index.html.
11. Smits, B., An RGB-to-Spectrum Conversion for Reflectances, Univ. of Utah, January 21, 2000.
12. Hoffmann and Gernot, *CIE Color Space*, 2000. www.fho-emden.de/~hoffmann/ciexyz29082000.pdf.