Spectral caustic rendering of a homogeneous caustic object based on wavelength clustering and eye sensitivity

Seah, Hock Soon; Lin, Feng; Tandianus, Budianto; Johan, Henry

2014

Tandianus, B., Johan, H., Seah, H. S., & Feng, L. (2014). Spectral caustic rendering of a homogeneous caustic object based on wavelength clustering and eye sensitivity. *The visual computer*, 31(12), 1601–1614.

<https://hdl.handle.net/10356/107170>

<https://doi.org/10.1007/s00371-014-1037-z>

© 2014 Springer-Verlag Berlin Heidelberg. This is the author created version of a work that has been peer reviewed and accepted for publication by *The Visual Computer*, Springer-Verlag Berlin Heidelberg. It incorporates referee's comments but changes resulting from the publishing process, such as copyediting, structural formatting, may not be reflected in this document. The published version is available at: [Article DOI: <http://dx.doi.org/10.1007/s00371-014-1037-z>].

Downloaded on 26 Apr 2023 18:00:30 SGT

Spectral caustic rendering of a homogeneous caustic object based on wavelength clustering and eye sensitivity

Budianto Tandianus · Henry Johan ·
Hock Soon Seah · Feng Lin

© Springer-Verlag Berlin Heidelberg 2014

Abstract In the real world, the index of refraction of a refractive object (caustic object) varies across the wavelengths. Therefore, in physically based caustic rendering, we need to take into account spectral information. However, this may lead to prohibitive running time. In response, we propose a two-step acceleration scheme for spectral caustic rendering. Our acceleration scheme takes into account information across visible wavelengths of the scene, that is, the index of refraction (IOR) (caustic object), light power (light), and material reflectance (surface). To process visible wavelengths effectively, firstly we cluster the wavelengths which have similar first refraction (air to caustic object) directions. In this way, all the wavelengths in a cluster can be represented by one light ray during rendering. Secondly, by considering the surrounding objects (their material reflectance from and visible surface area of the caustic objects) and light power, we com-

pute the refinement amount of each wavelength cluster. Our accelerated algorithm can produce photorealistic rendering results close to their reference images (which are generated by rendering every 1 nm of visible wavelengths) with a significant acceleration magnitude. Computational experiment results and comparative analyses are reported in the paper.

Keywords Rendering · Caustics · Spectral · Progressive photon mapping

1 Introduction

In the real world, light emitted from light sources consists of varying power over the wavelengths. This distribution of the light power over the wavelengths is commonly known as spectral power distribution (SPD). As a result, for a same scene, when we change the lights (each with different SPD), we will observe different colors. Similarly, each refractive object (caustic object) has a different property for each wavelength, which is its index of refraction. In many implementations, it is common to only consider wavelengths that correspond to red, green, and blue colors during rendering. However, as shown by Lai and Christensen [1], this approach is not able to produce accurate results. Thus, it is necessary to take into account the visible wavelengths (from 380 to 780 nm) for physically based rendering (spectral rendering).

A straightforward way to perform spectral caustic rendering is to render the interaction between densely sampled wavelength with caustic objects and combine the results (we call this a “brute force method”). The obvious problem of this approach is its high computational cost. One recent implementation of spectral caustics is provided by Hachisuka [2]. He performs spectral caustic rendering using the stochastic progressive photon mapping (SPPM) technique [3]. In his implementation, he randomly chooses a visible wavelength

B. Tandianus (✉)
gameLAB, School of Computer Engineering Organization (SCE),
Nanyang Technological University (NTU), Block NS4,
N4-B1b-13, 36 Nanyang Avenue, Singapore 639798, Singapore
e-mail: btandianus@ntu.edu.sg; budi0010@ntu.edu.sg

H. Johan
Fraunhofer IDM@NTU, Block NS1, N1-05-01, Nanyang Avenue,
Singapore 639798, Singapore
e-mail: henryjohan@ntu.edu.sg

H. S. Seah · F. Lin
School of Computer Engineering Organization (SCE),
Nanyang Technological University (NTU), Block NS4, N4-02a-32,
Nanyang Avenue, Singapore 639798, Singapore
e-mail: ashseah@ntu.edu.sg

F. Lin
School of Computer Engineering Organization (SCE),
Nanyang Technological University (NTU), Block NS4, N4-02a-32,
Nanyang Avenue, Singapore 639798, Singapore
e-mail: asflin@ntu.edu.sg

(a) Scene configuration

(b) Brute Force, F10 light

(c) Random, F10 light

Fig. 1 Comparison of brute force and Toshiya’s random sampling method. **a** Scene configuration. **b** Brute force. **c** Random sampling. All the experiments are done with the same total number of 802,000 iterations

to be processed in each SPPM iteration. By doing this, he assumes that all wavelengths share the same statistical information (i.e. gathering radius and flux). However, by doing so, the wavelength that is chosen randomly in the beginning will have different amount of contribution compared to the wavelengths chosen in subsequent iterations.

In Fig. 1, we show a comparison between SPPM brute force rendering and SPPM rendering with random wavelength selection (Hachisuka’s approach). The implementations of both approaches are based on Toshiya’s implementation [2]. The renderings were done using the F10 light source [4]. As we can see in the figure, the rendering results based on Hachisuka’s random wavelength sampling are generally noisier due to several peaks in the light SPD that are chosen only a few times in the random wavelength selection.

We provide a brief overview of existing works in spectral rendering in Sect. 2. In our work, we focus on spectral caustic effect and we aim to accelerate spectral caustic rendering while maintaining its rendering quality. Specifically, we render spectral caustics effects on a surface due to the light splitting event (over the spectral domain) when the light encounters a caustic object. The caustic object is assumed to

be homogeneous and non-scattering (for the subsurface scattering rendering work, readers are advised to check related literatures such as the work by Frisvad et al. for heavily scattering materials [5] and the work by Gutierrez et al. for thinly scattering materials such as underwater sea [6]). Our work supports scenes that have one or more caustic objects having the same index of refraction. One of the main applications of our work is to assist caustic object design and visualization. The designer can design the caustic object (e.g. a gemstone, crystal vase, glass bracelet, crystal pendant, glass chandelier, etc.) and use our method to render it to observe the refraction and caustics effects, which is similar to the application of Guy and Solers work [7].

We propose a two-step acceleration scheme for spectral caustics rendering. The first step of the acceleration scheme takes into account caustic object characteristic, i.e. the IOR over the wavelength $\eta(\lambda)$. Our basic idea is that the refraction angles of some wavelengths can be similar (within a certain threshold). Thus, we cluster those wavelengths into a single cluster and represent them as a single light ray in the rendering. In the second step, we take into account the spectral power distribution (SPD) of the light source, the surface reflectance in the scene, and the eye’s sensitivities toward light stimulus of each visible wavelength [8] to determine the importance level of each wavelength cluster obtained from the first step. Moreover, we also take into account a geometrical factor, i.e. the surface area of each material where caustic patterns might be generated. We then adaptively allocate resources for rendering each wavelength cluster based on its importance level. By doing this, we cut down the rendering time of wavelength clusters that contribute less to the final rendered image.

We perform the rendering using the stochastic progressive photon mapping (SPPM) technique [3]. The main reason for choosing SPPM is its iterative refinement nature which allows us to adaptively adjust the number of iterations for rendering each wavelength cluster. Another important reason is that SPPM does not require large memory footprint compared to the traditional photon mapping technique [9]. From the experimental results, our scheme can accelerate the rendering significantly while preserving its rendering quality. We compare the rendering results using our acceleration scheme against the Brute Force rendering result in Sect. 4. Finally, we draw conclusion and present our future work in Sect. 5.

2 Related work

Spectral rendering takes into account the spectral information of densely sampled visible wavelengths and thus it produces more accurate results compared to the general RGB-based rendering [10, 11]. The wavelength factor incurs costs in performance (processing every visible wavelength) and storage (storing the result for each visible wavelength). Due

to these factors, RGB-based rendering is more commonly used. For example, Shi et al. convert the necessary spectral information to RGB representation when they render polluted water [12]. Dong [13] proposed a rendering framework for a scene mixed with different material representation, i.e. spectral (for objects with thin films) and RGB (for objects with simple materials that can be rendered using a traditional RGB approach).

By taking spectral information into account, interesting effects, such as light interference on a thin layer [14–21], diffraction [22–24], chromatic lens aberrations [25], and rainbow [26,27], can be produced. Another interesting work by Xing et al. [28] can perform relighting of photographs. Spectral information is also useful in volume rendering [29,30].

The bottleneck in spectral rendering is the integration of the information over the visible wavelengths. There are several existing works in optimizing this computation. For example, the spectral information can be compressed using a linear transformation over a set of linear basis functions [31]. Also, it is possible to compress the spectral information using the wavelet transform [32,33]. Afterwards, we can progressively integrate from one level to another level. Xu and Sun compress the surface BRDF using Fourier transform and spherical harmonics [34]. Another approach is to minimize the spectral information by simplifying it while keeping the error to a minimum. For example, Zeghers et al. suppress/simplify spectral information over nearby wavelengths by connecting their two suppressed local peaks [35], Iehl and Peroch [36] represent a range of wavelengths as a constant value.

We can also optimize the computation by performing an analysis on the spectral information and perform subdivision or decomposition as necessary. For example, Deville et al.’s approach (which is the improvement of Meyer’s work [37]) takes into account the discontinuities and peaks in the spectral domain and they subdivide the spectral information so that it is possible to integrate analytically each of the subdivided range [38]. Meanwhile, Sun et al. take a different approach by decomposing the spectral information into smooth parts (that are transformed using Fourier basis) and spiky parts [39,40]. They consider the spiky parts as light sources such as CIE Standard Illuminant F11 which have peaks in some of the wavelengths. If they compress solely using Fourier basis, then the spiky parts will incur large errors.

Radziszewski et al. define a probability distribution function of the visible wavelengths based on their trial-and-error experiments [11]. They use the probability distribution function to sample wavelengths to be combined into a cluster in their photon mapping-based spectral rendering. However, they do not take into account light SPD, material reflectance and the IOR. Instead of using a trial-and-error probability distribution function, Evans and McCool sample a set of stratified random wavelengths [41] based on the SPD of the light source and put them inside a cluster. In this case, they do not

take into account the material property of the caustic object. As a result, light rays with wide angular refraction difference might be grouped into a cluster.

It is also possible to perform spectral rendering using photon mapping. However, the storage of the photons will become a big issue since we have to store the photons of all wavelengths. Lai and Christensen shoot photons for each wavelength, and they store the photons in RGB representation [1]. During photon gathering, they convert the photons energy back from RGB to spectral. However, due to the metamerism phenomenon, the conversion from RGB to spectral might not yield correct results. In a spectral rendering framework proposed by Iehl and Peroch [42], they shoot and deposit photons with randomly chosen wavelengths. They also save the first intersection of these photons with caustic objects. In the rendering during photon gathering, if the nearby visible point contains these photons, they then spawn photons from the first intersection but with different wavelengths.

In the GPU implementation [2] of the SPPM [3] by Hachisuka, he randomly selects a wavelength for photon shooting in each iteration and converts the photons into RGB format for photon gathering. In addition, the input for material reflectance is in RGB format. When a photon hits the surface, he converts the material reflectance from RGB to spectral representation using Smit’s proposed technique [43]. He also approximates the CIE XYZ using gaussian quadratures [44], hence he does not need to save the eye response function for each wavelength.

It is also worth mentioning that our proposed clustering is similar to the recent trend in ray tracing acceleration by packeting coherent light rays [45]. The main advantage of packeted light rays approach is that we can reduce the intersection test of all light rays in a packet to a single light ray. Basically, the ray packeting approach exploits coherency of the light ray direction in the spatial domain, whereas our acceleration scheme exploits the IOR similarity in the spectral domain.

Almost concurrent with our submission, Elek et al. propose a spectral ray differential rendering technique [46]. Similar to our proposed technique, Elek et al. also consider the difference of refraction direction with respect to wavelength changes. They use the difference in refraction direction as photon spread/differential instead of using the emitted light spread (as originally proposed by Schjth et al. [47]). By applying their technique to an existing approximate caustic rendering technique (proposed by Wyman and Davis [48]), they can achieve real-time rendering. Due to the nature of their proposed spectral differential (1D spread over the wavelength for each photon), their caustic reconstruction result exhibits line patterns. Their work has some differences with our proposed technique as follows. Firstly, our approach is based on ray packeting (clustering wavelengths

based on angular refraction similarity) and our result does not exhibit line patterns. In our formulation, we use angular refraction difference instead of refraction direction difference. Note that in this paper our acceleration schemes are applied to an offline renderer. Similar to Elek et al.’s work, our acceleration schemes can also be applied to a real-time approximate caustic renderer. It is possible to combine our work with Elek et al.’s work. For example, we cluster the wavelengths and pass this information to Elek et al.’s spectral differential for deciding which wavelengths to sample. Another possibility is using our second acceleration scheme to decide the amount of resources (such as number of photons) in the rendering of each randomly sampled wavelength in Elek et al.’s spectral ray differential renderer.

For more review on spectral work, readers are advised to read Devlin et al. state-of-the-art report on spectral rendering [49]. Their report covers various spectral rendering techniques and systems as well as visual effects that are generated with spectral renderers such as polarization and fluorescence effects. As noted by Devlin et al., there were still limited spectral renderer systems as of 2002. From some post-2002 related work we provide in this section, we can see that spectral rendering work is still growing.

3 Proposed spectral caustic rendering

The basic idea of our spectral caustic rendering is as follows. We first cluster the visible wavelengths, based on their refractive behaviors with respect to the caustic object, into several clusters (first acceleration step, Sect. 3.1). Then, for each cluster, we determine its refinement amount, based on the perceptual level and geometrical factor of surrounding surfaces, and decide the number of refinement iterations during the rendering (second acceleration step, Sect. 3.2). We then perform spectral caustic rendering using the SPPM technique [3] (please refer to Sect. 1 for the reasons of choosing SPPM). Since we only consider caustic objects during the clustering, in each rendering iteration, each photon will store the same amount of information and will perform a similar amount of workload. This factor is beneficial to GPU which favors coherence workload between threads.

We perform SPPM rendering for each wavelength cluster sequentially with each rendering using its computed number of refinement iterations. In the rendering, every photon carries the power of all wavelengths of the current cluster. When a photon hits a diffuse surface, we multiply the photon power with the reflectance of the surface for all the wavelengths in the cluster. On the other hand, when a photon hits a caustic object, we refract the photon based on the average IOR of the wavelengths in the current cluster. On the final SPPM iteration of a cluster, we multiply the final flux (of all the wavelengths of the current cluster) with CIE XYZ and convert them into RGB. Finally, we combine the rendering

results of all clusters by integrating their RGB results and produce the final image.

3.1 First acceleration step

The input to our wavelength clustering is the IOR curve of the caustic object. This information can be obtained from a real-world material or can be designed by users. In general, the IORs of real-world materials are monotonously decreasing with respect to wavelengths. Figure 2 shows an example of IOR values (y -axis) of a real-world material N-SF66 [50] with respect to wavelengths (x -axis). This information is obtained from the Schott Data Sheet [50]. We also take into account the materials with non-monotonously decreasing IOR, such as real-world materials which have absorption properties [51] (we show an example of IOR and its rendering result in Fig. 3) or artificial materials designed by users (Fig. 4a, b). For these materials, the curve can be of any shape. Thus, we first sort the wavelengths with decreasing IORs before we perform clustering. This is important since some wavelengths (that

Fig. 2 Plot of IOR of a real-world material N-SF66, with the x -axis representing wavelengths λ (in nm) and the y -axis representing IOR η . We also show the clustering results, with each cluster denoted by a *color line*. Note that in the areas where the changes of IOR are rapid, the clusters have less number of wavelengths (*smaller clusters*). For visualization purpose, we use a higher threshold value in this plot, i.e. 0.1, and the total number of clusters is 17

Fig. 3 IOR and a rendering result of a diamond caustic object with Rose Bengal 10 % solution material. **a** IOR. **b** Rendering result

Fig. 4 User’s designed IOR curves used in our experiments. **a** Clustered into 202 clusters. **b** Clustered into 204 clusters

are not next to each other) might refract the incoming light to the same direction. For instance, $\eta = 2$ in Fig. 4a intersects the IOR curve five times. Hence, by sorting the wavelengths based on their IORs before clustering, we can cluster non-neighboring wavelengths with similar refraction behaviors together and as a result we can obtain less number of clusters. For example, if we cluster the artificial IOR curves used in our experiments (shown in Fig. 4) without sorting, we obtain 247 clusters for the two curves. On the other hand, if we first sort the wavelengths, we obtain less number of clusters, 202 clusters for Fig. 4a and 204 clusters for Fig. 4b.

The clustering is done by firstly computing the absolute amount of change of refraction angle with respect to the change of wavelength $|d\theta/d\lambda|$ for all possible incoming angles for each 1 nm of visible wavelengths. Next, each wavelength is considered as an individual cluster, and then we group neighboring wavelengths if their sums of change of refraction angles are below a threshold h .

Change of refraction angle It can be computed as follows. We start with Snell’s law in Eq. 1.

$$\eta_2(\lambda) \sin(\theta_2) = \eta_1(\lambda) \sin(\theta_1), \quad (1)$$

where η_1 is the IOR of air, η_2 the IOR of caustic object, θ_1 the incoming angle and θ_2 the refraction angle. Both η_1 and η_2 are functions of wavelength λ since we take into account varying IOR across the wavelengths. The equation is then rearranged since we are interested in computing the refraction angle θ_2 . Hence, the refraction angle θ_2 is a function of the incoming angle θ_1 and wavelength λ . We assume that the IOR of air to be constant over the wavelength, i.e. 1.0. The rearranged equation is shown in Eq. 2.

$$\theta_2(\theta_1, \lambda) = \sin^{-1} \left(\frac{\sin(\theta_1)}{\eta_2(\lambda)} \right). \quad (2)$$

We can then compute the derivative of Eq. 2 with respect to the wavelength using the chain rule and we obtain the result as shown in Eq. 3.

$$\frac{d\theta_2(\lambda, \theta_1)}{d\lambda} = \frac{-\sin(\theta_1)\eta_2'(\lambda)}{\sqrt{\eta_2(\lambda)^2(\eta_2(\lambda)^2 - \sin(\theta_1)^2)}}, \quad (3)$$

where $\eta_2'(\lambda)$ is the first derivative of the caustic object’s IOR with respect to wavelength. As we are only interested in the

Fig. 5 Example plot of $|d\theta_2(\lambda, \theta_1)/d\lambda|$ (in degree/nm) for a real-world material N-SF66. The θ_1 -axis represents the incoming angle (in degree) and the λ -axis represents wavelength (in nm)

amount of change, irrespective of negative or positive change, we take the absolute value of Eq. 3.

We show an example plot $|d\theta_2(\lambda, \theta_1)/d\lambda|$ of a real-world material N-SF66 [50] in Fig. 5. The total amount of changes in refraction angle for a wavelength $\Theta(\lambda)$ is computed as the sum of changes for incoming angles (from 0° to 89°), as shown in Eq. 4.

$$\Theta(\lambda) = \int_0^{89^\circ} \left| \frac{d\theta_2(\lambda, \theta_1)}{d\lambda} \right| d\theta_1. \quad (4)$$

Referring to the example in Fig. 5, we compute the volume below the surface.

Clustering We consider every 1 nm wavelength between 380 and 780 nm, that is, in total we have 401 wavelengths. We perform the clustering by iteratively merging two neighboring wavelength clusters whose sum of their $\Theta(\lambda)$ is below a threshold h . The clustering process is shown in Algorithm 1. The threshold h is determined based on the maximum allowable change of refraction angle α with respect to the change of wavelength. Using Eq. 4, the maximum allowable $\Theta(\lambda)$ for merging clusters is $h = \int \alpha d\theta_1 = 89\alpha$. In practice, we

Algorithm 1: Clustering Algorithm

```

1 for each of the 401 wavelengths (380 nm to 780 nm) do
2   if it's  $\Theta(\lambda) \geq h$  then
3     Mark it as a final cluster;
4   else
5     Mark it as a free cluster;
6   end
7 end
8 while there are free clusters do
9   Find a pair of neighboring free clusters whose sum of their
    $\Theta(\lambda)$  is the smallest;
10  if the sum  $\leq h$  then
11    Merge the two clusters, assign the sum as  $\Theta(\lambda)$ , and mark
    the new cluster as a free cluster;
12  else
13    Mark all the free clusters as final clusters;
14  end
15 end

```

Fig. 6 Image differences between rendering results with varying threshold value, with *brighter pixels* indicating larger differences. Each cluster is rendered with 2,000 iterations. For each pixel, we compute the Euclidean distance and we scale it by 10.0. The first, second, and third columns show image differences between 0.5° and 0.1° , 0.1° and 0.05° , 0.05° and 0.01° , respectively. We use N-SF66 (Fig. 2), Arbitrary 2 (Fig. 4b), and Arbitrary 1 (Fig. 4a) IOR curves on the caustic objects of the first, second, and third row, respectively. As for the light source, we use CIE Standard illuminant F10 for the first row, and D65 for the second and third rows. The scene configuration of the first row is shown

set α to 0.1° . We determine this value from various experiments, and we show one of them in Fig. 6. The scene is illuminated with CIE Standard Illuminant D65 and the IOR of the caustic object is Arbitrary 1 (Fig. 4a). We use Arbitrary

in Fig. 7. The light source of the scenes on the second row is a point light source directly above the caustic object and on the third row is a point light source directly below the caustic object. **a** 0.5° (4 clusters) vs. 0.1° (17 clusters). **b** 0.1° (17 clusters) vs. 0.05° (34 clusters). **c** 0.05° (34 clusters) and 0.01° (137 clusters). **d** 0.5° (58 clusters) vs. 0.1° (152 clusters). **e** 0.1° (152 clusters) vs. 0.05° (204 clusters). **f** 0.05° (204 clusters) and 0.01° (346 clusters). **g** 0.5° (55 clusters) vs. 0.1° (147 clusters). **h** 0.1° (147 clusters) vs. 0.05° (202 clusters). **i** 0.05° (202 clusters) vs. 0.01° (345 clusters)

1 IOR as an example due to its non-monotonous property that is suitable for testing our first acceleration step. Figure 2 shows the clustering results of a real-world material N-SF66.

Fig. 7 The overall scene configuration of the results shown on the first row of Figs. 6, 10 and 14. The light source is a point light source located near the face of the bunny

3.2 Second acceleration step

Generally, we perceive every visible wavelength arriving at our retina based on the emitted spectral power distribution (SPD) of the light source, the surface reflectance of materials in the scene, and our eyes' sensitivity for each visible wavelength. By taking into account these information, we propose an importance level $f(\lambda)$ that controls the number of rendering refinement for each wavelength cluster. That is, for clusters with lower $f(\lambda)$ values, we refine less times compared to clusters with higher $f(\lambda)$ values. We show the formula in the following Eq. 5

$$f(\lambda) = l(\lambda)y(\lambda) \sum_i m_i(\lambda)L_i\hat{g}_i, \quad (5)$$

where $l(\lambda)$ is the light's SPD with respect to wavelength λ (we can use a real-world light source [4] or user's designed), $y(\lambda)$ the luminosity function, $m_i(\lambda)$ the i th material's surface reflectance with respect to wavelength λ (value in the range [0..1] (unit less) as it describes the portion of light that is reflected; we can use a real-world material such as the reflectance of Macbeth Color Checker [4] or that is designed by a user). For non-diffuse surfaces which have varying reflectance amount over the possible reflected directions, we handle these surfaces by taking their maximum reflectance for m_i . By doing this, we might overestimate but will not underestimate the reflectance. L_i is the weighting based on the lightness of the surface material m_i . The lightness is the same as the L^* of the CIE $L^*A^*B^*$ color space. $\hat{g}_i$ is the weight based on the surface area of material m_i visible from the caustic object.

Perceptual factor affects the refinement as it is redundant to use the same amount of resources for rendering surfaces that have different levels of perceptual. We incorporate two perceptual factors in the $f(\lambda)$ computation:

Fig. 8 We sample the material types of the surrounding surfaces from the bounding box (blue box) corners and the center of the caustic object. The numbers denote the material type. As shown in the figure, sampling of material #3 is higher as the smaller box enclosing the caustic object has material #3. Some parts of material #0 and #2 are sampled, and none of material #1 is sampled

- Luminosity function $y(\lambda)$ that describes the human eye sensitivity in the spectral domain (i.e. human eyes have different amounts of brightness sensation over the visible wavelength range). The luminosity function is the second CIE standard color matching function ($y(\lambda)$).
- Lightness L_i , the non-linear perceptual property of the material.

Moreover, we also consider the geometrical distribution of the surface material. More weight is given to materials whose surfaces receive more caustic patterns. To compute the weight $\hat{g}_i$, we use spherical uniform sampling to sample the surrounding of caustic object. The weight $\hat{g}_i$ is computed as $\hat{g}_i = g_i / \sum_j g_j$ (normalized to range [0..1]) where g_i is the number of sample points on the surfaces with material m_i . The sampling is illustrated in Fig. 8. We show our experiment on geometrical factor in Sect. 4.2.

The $f(\lambda)$ of all visible wavelengths are then normalized such that the highest $f(\lambda)$ has the value of 1.0. Figure 9 shows some examples of $f(\lambda)$. For comparison, we show the $f(\lambda)$ proposed by Radziszewski et al. [11] which is scene independent (always fixed) in Fig. 9a.

Based on $f(\lambda)$, we have several options to set the importance level of a wavelength cluster. For the first option, we use the maximum $f(\lambda)$ of the wavelengths in the cluster. The drawback of this approach is that a cluster dominated by wavelengths with low $f(\lambda)$ but has few wavelengths with very high $f(\lambda)$ will have too many rendering iterations. Another option is to use the average $f(\lambda)$ of the wavelengths in the cluster. Although a cluster dominated by low $f(\lambda)$ but has a few wavelengths with very high $f(\lambda)$ might not have enough rendering iterations, in practice, we adopt the former approach in order not to underestimate the refine-

Fig. 9 Plot of $f(\lambda)$ (y-axis) over the visible wavelengths λ (x-axis) for various scenes. **a** $f(\lambda)$ proposed by Radziszewski et al. It is fixed, independent of scenes. **b** Our $f(\lambda)$ for the scene in Fig. 14b. **c** Our $f(\lambda)$ for the scene in Fig. 14e. **d** Our $f(\lambda)$ for the scenes in Fig. 14h and k

ment iteration amount. The number of iterations for a wavelength cluster is then determined by multiplying its importance level with M , where M is the maximum number of iterations.

4 Experiments

We perform our experiments on a PC with an Intel i7-3820 3.60 GHz CPU and a NVIDIA GTX 690 4 GB. The rendering implementation is done using OptiX, a GPU-based ray tracing engine [52].

4.1 Brute force rendering

For the sake of fair comparisons between the Brute Force method and our method, we firstly perform experiments to find appropriate parameters for the Brute Force experiments. In this case, we try to determine M (maximum number of iterations) and wavelength nanometer step. We typically set $M = 2,000$ as we can achieve good rendering quality while keeping the rendering time manageable, especially in performing Brute Force rendering (i.e. we render $M = 2,000$ iterations for each 1 nm of visible wavelengths). As we show in Fig. 10, the image difference between 2,000 and 3,000 iterations is pretty low.

In the brute force rendering whose results are shown in Fig. 10, we use 1 nm step as we try to be as conservative as possible in order not to miss any signal variation that can contribute to the final rendering. To determine a good nanometer step of the wavelength in the Brute Force computation (and at the same time to provide fair comparison with our proposed acceleration scheme), we perform the following experiments. We render scenes with 401 clusters (each covers 1 nm), 200 clusters (each covers around 2 nm), and 100 clusters (each covers around 4 nm). We show the image difference results (between 100 and 200 clusters, 200 and 401

clusters) in Fig. 11 for two scenes. As shown in the results, the differences are smaller between 200 and 401 clusters. Moreover, due to the peaks in F10 SPD, the sampling in 100 clusters (every around 4 nm) is not sufficient as it affects the illumination of the whole scene (Fig. 11a). There are still noticeable differences between 200 and 401 clusters in Fig. 11d. Hence, in the following comparison experiments, we use 401 clusters (1 nm sampling step) for the brute force rendering.

4.2 Geometrical factor experiment

We show the comparison between considering and not considering the geometrical factor (i.e. for not considering the geometrical factor, the g_i is always 1.0) in Fig. 12. The scene setting is similar to the scene shown in Fig. 8, but with additional smaller blue walls (they have the same material as the left wall) enclosing the bunny. The only open side is the left side facing the blue wall, such that the caustic patterns can land on the left wall and on the floor. The scene arrangement is illustrated in Fig. 7. As seen in the results, by considering the geometrical factor, the rendering speed is slightly improved compared to not considering the geometrical factor.

4.3 Comparisons

Experimental results of our methods are shown in Fig. 14 with the IOR curves used in our experiments shown in Figs. 2 and 4. All images are rendered with a resolution of 512×512 . We show the comparisons between three spectral rendering methods.

The first method (results are shown on the first column) is the Brute Force rendering that performs SPPM rendering for each 1 nm of visible wavelengths (with $M = 2,000$ iterations for each 1 nm wavelength). The second method (results are shown on the second column), is the combination of our first and second acceleration steps. The third method (results are shown on the third column) uses our first acceleration step to cluster the wavelengths and then uses Radziszewski et al. probability distribution function $f(\lambda)$ to determine the number of iterations for each wavelength cluster [11]. The purpose of this comparison is to show the effectiveness of our second acceleration step.

Compared to the rendering time using the brute force method, the combination of our first and second acceleration steps can significantly accelerate the rendering (especially for the scene on the first row with a real-world material N-SF66 in which we can cluster the visible wavelengths to just 34 clusters; this shows the effectiveness of our first acceleration step) while preserving the image quality. Moreover, using our second acceleration step, we can perform the rendering faster compared to using Radziszewski et al.’s $f(\lambda)$.

Fig. 10 Image differences between Brute Force rendering results with varying iterations (in 1 nm step), with *brighter pixels* indicating larger differences. For each pixel, we compute the Euclidean distance and we scale it by 10.0. The first, second, and third columns show image differ-

ence between 500 and 1,000 iterations, 1,000 and 2,000 iterations, 2,000 and 3,000 iterations, respectively. The scene configuration of each row corresponds to the scene configuration of each row of Figs. 6 or 14

Overall, our acceleration scheme achieves higher acceleration on caustics object that has a smaller range of index of refraction (such as natural material, N-SF66) since we can obtain less number of clusters (34 clusters for N-SF66).

4.4 Discussions

The main concern of our proposed method is that the clustering in our first acceleration step only takes into account the

first refraction. The refracted clustered light might diverge further on its subsequent reflections and/or refractions. For instance, in the second intersection event, some wavelengths in the cluster should be reflected due to total internal reflection and some others should be refracted (exit the caustic object). However, in our acceleration scheme, we do not split the cluster further. We perform experiments to investigate this issue. We render caustics caused by a stack of cylinders (which has Arbitrary 2 IOR) and D65 light source. Due

Fig. 11 Image differences between brute force rendering results with varying nanometer steps, with *brighter pixels* indicating larger differences. For each pixel, we compute the Euclidean distance and we scale it by 10.0. **a** Image difference between 100 and 200 clusters, **b** image difference between 200 and 401 clusters, **c** image difference between

100 and 200 clusters, **d** image difference between 200 and 401 clusters. **(a, b)** The same scene as the scene in the first row of Fig. 10, with N-SF66 IOR (Fig. 2) and F10 light source. **(c, d)** The same scene as the scene in the second row of Fig. 10, with Arbitrary 2 IOR (Fig. 4b) and D65 light source

Fig. 12 Geometrical factor experiment result. We show the rendering result of experiment with geometrical factor as the caustic patterns can be rendered in the same quality as the rendering result without geometrical factor. The rendering with geometrical factor took 15.61 min and the rendering without geometrical factor took 16.18 min

to the placement of caustic objects, we expect the light to undergo several intersections. We show the rendering result in Fig. 13, where only very subtle differences in caustic patterns are observed. The same phenomenon is also observed for the diamond caustic object as shown in Fig. 14g and h. The light source is positioned between the floor and the diamond. As a result, the caustic patterns we see on the floor are generated from the lights that undergo several total internal reflections inside the diamond caustic object.

From our experiments, including the ones with materials that have a wide range of IOR, we can see that it is sufficient to consider only the first refraction in the wavelength clustering step. It is sufficient because the light ray energy decreases if it undergoes multiple reflections and/or

(a)

(b)

Fig. 13 Rendering comparison of caustics that is produced by multiple refractions. **a** Brute force approach, **b** our approach

refractions and thus it will not affect the final caustics much. Hence, in practice, especially spectral caustic rendering of real-world caustic objects that have a smaller range of

Fig. 14 Our experimental results. The results in each column are generated using different methods. The first column is generated using the brute force method (2,000 iterations, with 1 nm step), the second column is generated using our method, and the third column is generated using our method but for the second acceleration step we use the probability distribution function proposed by Radziszewski et al. We use N-SF66 (Fig. 2), Arbitrary 2 (Fig. 4b), Arbitrary 1 (Fig. 4a), and Rose Bengal solution (Fig. 3a) IOR curves on the caustic objects of the first, second, and third, and fourth rows, respectively. **a** 38.62 h, 802,000 iterations, **b** 0.28 h (speed up 137.93 times), 5,088 iterations, **c** 2.31 h (speed up 16.72 times), 47,341 iterations, **d** 37.95 h, 802,000 iterations, **e** 2.99 h (speed up 12.69 times), 60,301 iterations, **f** 11.10 h (speed up 3.42 times), 233,106 iterations, **g** 33.57 h, 802,000 iterations, **h** 3.46 h (speed up 9.70 times), 78,714 iterations, **i** 9.90 h (speed up 3.39 times), 231,239 iterations, **j** 33.96 h, 802,000 iterations, **k** 1.94 h (speed up 17.51 times), 37,699 iterations, **l** 2.12 h (speed up 16.02 times), 41,572 iterations

IOR, our wavelength clustering is sufficient to produce good results.

5 Conclusion and future work

We have presented a two-step scheme for accelerating spectral caustics rendering. Our experimental results show that

our proposed scheme can achieve rendering quality close to the brute force rendering. Moreover, the range of our rendering speed acceleration (comparing to the rendering speed using brute force) magnitude is from tens to hundreds. We also compare our second acceleration step with the scene independent probability distribution function $f(\lambda)$ (for computing refinement iterations) proposed by Radziszewski et al. [11].

Our first acceleration step only takes into account the characteristic of a caustic object. As such, this work can be improved for handling more complicated scenes that have multiple caustic objects with different index of refractions. It is also possible to do clustering using another approach, i.e. wavelength clustering of each photon taking into account its incoming angle onto the caustic object surface. This approach requires additional bookkeeping in the process. For example, each photon will store a different number of wavelength clusters. Thus, the integration computation for each photon will involve different number of iterations. In GPU implementation, it is a good practice for each thread to do the same amount of workload and as such, it needs an in-depth investigation of the efficiency of this new clustering approach. We intend to try this alternative clustering approach and compare it with our current approach in the future.

In the second acceleration step, we use uniform sampling to sample material types surrounding the caustic object. However, the sampling may sample surfaces that are not hit by photons. It can be improved using a real-time approximate caustic renderer to do the sampling for more accurate results.

For a very special case such as Newton's prism experiment with a receiver plane that is very tangential to its refraction direction, the generated rainbow pattern may stretch widely. Even if we render using the Brute Force approach, discontinuity in the rainbow pattern can be observed. One possible approach to remove such discontinuity is by combining our work with Elek et al.'s spectral ray differential [46].

We are also interested to apply our acceleration scheme to other rendering algorithms, such as bi-directional path tracing. In this case, we cluster the rays of several wavelengths using our first acceleration step, and then vary the number of sample per pixels (or path lengths) using our second acceleration step.

Acknowledgments This work is partially supported by a research grant MOE2011-T2-2-037 from Ministry of Education, Singapore. Henry Johan is supported by Fraunhofer IDM@NTU, which is funded by the National Research Foundation (NRF) and managed through the multiagency Interactive & Digital Media Programme Office (IDMPO) hosted by the Media Development Authority of Singapore (MDA).

References

1. Lai, G., Christensen, N.: A compression method for spectral photon map rendering. In: 15th International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision 2007, WSCG'2007 - In Co-operation with EUROGRAPHICS, Full Papers Proceedings WSCG Proceedings, pp. 95–102. Plzen (2007)
2. Hachisuka, T.: Toshiya Hachisuka at UTokyo <http://www.ci.i.u-tokyo.ac.jp/hachisuka/>
3. Hachisuka, T., Jensen, H.W.: Stochastic progressive photon mapping. *ACM Trans. Graph.* **28**(5), 141:1–141:8 (2009)
4. Useful color data. <http://www.cis.rit.edu/research/mcsl2/online/cie.php>
5. Frisvad, J.R., Christensen, N.J., Jensen, H.W.: Computing the scattering properties of participating media using lorenz-mie theory. *ACM Trans. Graph.* **26**(3) (2007). doi:[10.1145/1276377.1276452](https://doi.org/10.1145/1276377.1276452)
6. Gutierrez, D., Seron, F.J., Munoz, A., Anson, O.: Visualizing underwater ocean optics. *Comput. Graph. Forum* **27**(2), 547–556 (2008). doi:[10.1111/j.1467-8659.2008.01152.x](https://doi.org/10.1111/j.1467-8659.2008.01152.x)
7. Guy, S., Soler, C.: Graphics gems revisited: fast and physically-based rendering of gemstones. *ACM Trans. Graph.* **23**(3), 231–238 (2004). doi:[10.1145/1015706.1015708](https://doi.org/10.1145/1015706.1015708)
8. Wright, W.D.: A re-determination of the trichromatic coefficients of the spectral colours. *Trans. Opt. Soc.* **30**(4), 141 (1929)
9. Jensen, H.W.: Global illumination using photon maps pp. 21–30 (1996)
10. Johnson, G., Fairchild, M.: Full-spectral color calculations in realistic image synthesis. *IEEE Comput. Graph. Appl.* **19**(4), 47–53 (1999)
11. Radziszewski, M., Boryczko, K., Alda, W.: An improved technique for full spectral rendering. *J. WSCG* **17**(1–3), 9–16 (2009)
12. Shi, J., Zhu, D., Zhang, Y., Wang, Z.: Realistically rendering polluted water. *Vis. Comput.* **28**(6–8), 647–656 (2012). doi:[10.1007/s00371-012-0685-0](https://doi.org/10.1007/s00371-012-0685-0)
13. Dong, W.: Rendering optical effects based on spectra representation in complex scenes. In: Nishita, T., Peng, Q., Seidel, H.-P. (eds.) *Advances in Computer Graphics. Lecture Notes in Computer Science*, vol. 4035, pp. 719–726. Springer, Berlin Heidelberg (2006)
14. Gondek, J.S., Meyer, G.W., Newman, J.G.: Wavelength dependent reflectance functions. Proceedings of the 21st annual conference on Computer graphics and interactive techniques. SIGGRAPH '94, pp. 213–220. ACM, New York, NY, USA (1994)
15. Hirayama, H., Kaneda, K., Yamashita, H., Yamaji, Y., Monden, Y.: Visualization of optical phenomena caused by multilayer films with complex refractive indices. In: *Computer graphics and applications*, 1999. Proceedings. Seventh Pacific Conference on, pp. 128–137, 320 (1999)
16. Sun, Y., Fracchia, F., Calvert, T., Drew, M.: Deriving spectra from colors and rendering light interference. *Comput. Graph. Appl. IEEE* **19**(4), 61–67 (1999)
17. Iwasaki, K., Matsuzawa, K., Nishita, T.: Real-time rendering of soap bubbles taking into account light interference. In: *Proceedings of the Computer graphics international*, pp. 344–348. IEEE Computer Society (2004)
18. Duriković, R., Kimura, R.: Spectrum-based rendering using programmable graphics hardware. In: *Proceedings of the 21st spring conference on Computer graphics*, pp. 233–236. ACM, Budmerice, Slovakia (2005)
19. Durikovic, R., Kimura, R.: GPU rendering of the thin film on paints with full spectrum. In: *Information visualization, 2006. IV 2006. Tenth International Conference on*, pp. 751–756 (2006)
20. Sun, Y.: Rendering biological iridescences with RGB-based renderers. *ACM Trans. Graph.* **25**(1), 100–129 (2006)
21. Sun, Y., Wang, Q.: Interference shaders of thin films. *Comput. Graph. Forum* **27**, 1607–1631 (2008)
22. Stam, J.: Diffraction shaders. In: *Proceedings of the 26th annual conference on Computer graphics and interactive techniques*, pp. 101–110. ACM Press/Addison-Wesley Publishing Co. (1999)
23. Agu, E.O.: Diffraction shading models in computer graphics. Ph.D. thesis, University of Massachusetts Amherst (2001)
24. Imura, M., Abe, T., Kanaya, I., Yasumuro, Y., Manabe, Y., Chihara, K.: Rendering of Play of color using stratified model based on amorphous structure of opal. *Techniques and Applications, Proceedings VIIth Digital Image Computing* (2003)
25. Wu, J., Zheng, C., Hu, X., Xu, F.: Rendering realistic spectral bokeh due to lens stops and aberrations. *Vis. Comput.* **29**(1), 41–52 (2013). doi:[10.1007/s00371-012-0673-4](https://doi.org/10.1007/s00371-012-0673-4)

26. Musgrave, F.K.: Prisms and rainbows: a dispersion model for computer graphics. *Proc. Graph. Interface* **89**, 227–234 (1989)
27. Sadeghi, I., Munoz, A., Laven, P., Jarosz, W., Seron, F., Gutierrez, D., Jensen, H.W.: Physically-based simulation of rainbows. *ACM Trans. Graph.* **31**(1), 3:13:12 (2012). doi:[10.1145/2077341.2077344](https://doi.org/10.1145/2077341.2077344)
28. Xing, X., Dong, W., Zhang, X., Paul, J.C.: Spectrally-based single image relighting. In: *Proceedings of the Entertainment for education and 5th international conference on E-learning and games, Edutainment'10*, pp. 509–517. Springer, Berlin, Heidelberg (2010)
29. Bergner, S., Mller, T., Drew, M.S., Finlayson, G.D.: Interactive spectral volume rendering. In: *Proceedings of the conference on Visualization '02*, pp. 101–108. IEEE Computer Society, Boston, Massachusetts (2002)
30. Strengert, M., Klein, T., Botchen, R., Stegmaier, S., Chen, M., Ertl, T.: Spectral volume rendering using GPU-based raycasting. *Vis. Comput.* **22**(8), 550–561 (2006)
31. Marimont, D.H., Wandell, B.A.: Linear models of surface and illuminant spectra. *J. Opt. Soc. Am. A* **9**(11), 1905–1913 (1992)
32. Chern, J.R., Wang, C.M.: A novel progressive refinement algorithm for full spectral rendering. *Real Time Imaging* **11**(2), 117–127 (2005)
33. Rougeron, G., Proche, B., Lisse, C.S.: An adaptive representation of spectral data for reflectance computations. In: *Eurographics Rendering Workshop*, pp. 127–138. Springer (1997)
34. Xu, H., Sun, Y.: Compact representation of spectral BRDFs using fourier transform and spherical harmonic expansion. *Comput. Graph. Forum* **25**(4), 759–775 (2006)
35. Zeghers, E., Carr, S., Bouatouch, K.: Error-bound wavelength selection for spectral rendering. *Vis. Comput.* **13**(9–10), 424–434 (1998)
36. Iehl, J., Proche, B.: Adaptive spectral rendering with a perceptual control. *Comput. Graph. Forum* **19**(3), 291–299 (2000)
37. Meyer, G.W.: Wavelength selection for synthetic image generation. *Comput. Vis. Graph. Image Process.* **41**(1), 57–79 (1988)
38. Deville, P.M., Merzouk, S., Cazier, D., Paul, J.C.: Spectral data modeling for a lighting application. *Comput. Graph. Forum* **13**(3), 97–106 (1994)
39. Sun, Y., Fracchia, F.D., Drew, M.S.: A composite spectral model and its applications. pp. 102–107 (2000)
40. Sun, Y., Fracchia, F.D., Drew, M.S., Calvert, T.W.: A spectrally based framework for realistic image synthesis. *Vis. Comput.* **17**(7), 429–444 (2001)
41. Evans, G.F., McCool, M.D.: Stratified wavelength clusters for efficient spectral monte carlo rendering. In: *Proceedings of the 1999 conference on Graphics interface '99*, p. 42–49. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA (1999)
42. Iehl, J., Proche, B.: Towards perceptual control of physically based spectral rendering. *Comput. Graph. (Pergamon)* **27**(5), 747–762 (2003)
43. Smits, B.: An RGB-to-spectrum conversion for reflectances. *J. Graph. Tools* **4**(4), 11–22 (1999)
44. Wallis, R.: Fast computation of tristimulus values by use of gaussian quadrature. *J. Opt. Soc. Am.* **65**(1), 91–94 (1975)
45. Boulos, S., Edwards, D., Lacewell, J.D., Kniss, J., Kautz, J., Shirley, P., Wald, I.: Packet-based whitted and distribution ray tracing. In: *Proceedings of Graphics Interface 2007, GI '07*, p. 177–184. ACM, New York, NY, USA (2007). doi:[10.1145/1268517.1268547](https://doi.org/10.1145/1268517.1268547)
46. Elek, O., Bauszat, P., Ritschel, T., Magnor, M., Seidel, H.P.: Spectral ray differentials. *Computer Graphics Forum (Proceedings of EGSR)* **33**(4) (2014)
47. Schjøth, L., Frisvad, J.R., Erleben, K., Sporrang, J.: Photon differentials. In: *Proceedings of the 5th international conference on Computer graphics and interactive techniques in Australia and Southeast Asia*, pp. 179–186. ACM, Perth, Australia (2007). doi:[10.1145/1321261.1321293](https://doi.org/10.1145/1321261.1321293)
48. Wyman, C., Davis, S.: Interactive image-space techniques for approximating caustics. In: *Proceedings of the 2006 symposium on Interactive 3D graphics and games*, pp. 153–160. ACM, Redwood City, California (2006). doi:[10.1145/1111411.1111439](https://doi.org/10.1145/1111411.1111439)
49. Devlin, K., Chalmers, A., Wilkie, A., Purgathofer, W.: STAR: Tone reproduction and physically based spectral rendering. In: D. Feller, R. Scopigno (eds.) *State of the Art Reports, Eurographics 2002*, pp. 101–123. The Eurographics Association (2002)
50. Downloads | SCHOTT north america. http://www.us.schott.com/advanced_optics/english/download/index.html
51. Weidlich, A., Wilkie, A.: Anomalous dispersion in predictive rendering. *Comput. Graph. Forum* **28**(4), 1065–1072 (2009)
52. NVIDIA OptiX ray tracing engine. <http://developer.nvidia.com/optix>

Budianto Tandianus Budianto Tandianus is currently a Research Associate as well as a doctoral student in gameLAB, Nanyang Technological University (Singapore). He received Sarjana Komputer Degree in Computer Engineering from STMIK-Mikroskil (Indonesia) in 2005 and M.Sc. degree in Digital Media Technology from Nanyang Technological University (Singapore) in 2008. His research interests include rendering, animation, gaming technology, and multimedia.

Henry Johan is a Senior Research Fellow in Fraunhofer IDM@NTU (Singapore). Previously he was a Post-Doctoral Fellow in the Department of Complexity Science and Engineering at the University of Tokyo (Japan). Then, he joined the School of Computer Engineering at Nanyang Technological University (Singapore) as an Assistant Professor. His research interests in computer graphics include rendering, animation, and shape retrieval. He received his BS, MS, and PhD degrees in Computer Science from the University of Tokyo in 1999, 2001, and 2004, respectively.

Hock Soon Seah is a Professor and Director of the Multiplatform Game Innovation Centre (MAGIC) at the School of Computer Engineering, Nanyang Technological University. He has more than 20 years of experience in computer graphics and imaging research. His research interests are in geometric modeling, image sequence analysis with applications to digital film effects, automatic in-between frame generation from hand-drawn sketches, augmented

reality and advanced medical visualization.

Feng Lin is an Associate Professor in the School of Computer Engineering, Nanyang Technological University, and the Director of Bioinformatics Research Centre. His research interest includes biomedical informatics, bioimaging, computer graphics and visualization, and high-performance computing. He is a Senior Member of IEEE.