DOI: 10.1111/cgf.14397
Pacific Graphics 2021
E. Eisemann, K. Singh, and F.-L Zhang
(Guest Editors)

Volume 40 (2021), Number 7

Global Illumination-Aware Stylised Shading

K. Doi, Y. Morimoto, and R. Tsuruno

Abstract

Our aim is to convert an object's appearance to an arbitrary colour considering the light scattering in the entire scene, which is often called the global illumination. Existing stylisation methods convert the colour of an object with a 1-dimensional texture for 3-dimensional computer graphics to reproduce a typical style used in illustrations and cel animations. However, they cannot express global illumination effects. We propose two individual methods to compute the global illumination and convert the shading to an arbitrary colour. The methods reproduce reflections in other objects with the converted colour. As a result, we can convert the colour of illumination effects that have not yet been reproduced, such as soft shadows and refractions.

CCS Concepts

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

1. Introduction

In hand-painted illustrations, there are stylistic expressions that change the hue along with the brightness of an object, and others that quantise its colour. In illustrations, these expressions often appear with realistic illumination effects such as caustics and soft shadows, which depend on the entire scene. For example, knowledge of the relative position of the light sources, the target object, and any objects between them is required to render the shadow. There are cases for which global illumination should be considered, even for actual professional illustrations such as the one in Figure 1. In this illustration the hair colour of a character is reflected in the hair of the character next to it. Thus, considering global illumination in stylisation helps express the reflective nature of a material better by including other objects' colours on own surface.

For clarity, our purpose is to apply an arbitrary 1D texture considering global illumination to calculate of the light of a single pixel. We do not intend non-photorealistic rendering that uses 2D view space textures to represent painting style like oil paintings.

Many rendering methods have been proposed to create images in which stylised colour and photorealistic shading are mixed. Stylised shading methods [LMHB00, SMGG01, TAY13, MSGS13, FJL*16] convert calculated shading into step changes of a few colours that a user defines. Methods that use scene geometry [LMHB00, SMGG01, TAY13] lookup the input texture referring to the light or viewpoint from the object to generate hard shading with a few colours as found in hand-painted cel animations. The same effect can be achieved by postprocessing the rendered result [MSGS13, FJL*16]. Some functions for stylisation that remap the computed shadings are provided by common tools such as Blender [Ble]. However, it is difficult to reproduce the interactions of shading in the scene such as the colour bleeding of a converted colour into the surrounding objects. Therefore, these methods cannot be applied in a scene in which the light interactions are visually important, for example, the caustics caused by light passing through glass. As illumination editing for physically based rendering, path manipulation [SNM*13, GRR*16, ROTS09] and material design [GGT*20, NSRS13] have been proposed. However, those methods do not produce colour gradation corresponding to physically based shading, unlike the stylisation technique with texture input we explained above.

Figure 1: *Physical phenomena such as reflection or caustics even in non-photorealistic colour transition. The green colours appear in red hair. Comic cover drawn by Haruko Ichikawa [Ich15].*

We propose two colour remapping methods in this paper: *Colour Remapping that Affects the Colour on a Path Trace* (ACP), and

Fidelity for Texture Values (FTV). Both express the shading that varies in photorealistic shading using 1-dimensional (1D) texture, whereas they reproduce the scattering in surrounding objects by computing the global illumination.

ACP converts the estimated values of the rendering equation [Kaj86] on the object by sampling one incoming direction. The resulting shading is the sum of the values on the assigned texture obtained from each sample path.

FTV samples many incident paths to estimate the radiance, so the object radiates one colour on the texture. To reduce the computation, we convert the colour on exactly one vertex of the complete light path from the source to the eye. The implementation of FTV consists of subpath tracing from the eye to the object, and estimation of the radiance of the object using progressive photon mapping (PPM) [HOJ08]. We store and reuse the results to reduce the computation required for converting the radiance to an arbitrary colour.

The main contributions are as follows.

- Both colour remapping methods use a 1D texture and consider the reflection and refraction among objects.
- Comparisons show that both methods reproduce global illumination better than existing methods.
- In ACP, the converted colour affects the colour at the next intersection on a path traced path.
- In FTV, the resulting image only contains the specified input texture colours.
- FTV contains an effective PPM-based calculation that enables the object's radiance to be converted afterward.

2. Related Work

In Section 2.1, we explain why existing stylised shading methods do not reproduce global illumination effects. In Section 2.2, we review the rendering methods we apply to implement our methods.

2.1. Stylised Shading

In 3D computer graphics, to reproduce the colour tones used in illustrations, rendering results are remapped to specified input colours.

Many methods based on cel shading express characteristic shadings using the scene geometry to fix the colour. One method [LMHB00] computes the direct illumination using the object normal and the direction to the source, and determines its colour referring to a 1D texture. Lit Sphere [SMGG01] and its extension [TAY13] additionally use the viewing direction to map a 2D image on a sphere. As an extension of cel shading, Barla et al. proposed a method that controls the detail of the texture as well as its colour [BTM06]. Anjyo et al. [AWB06] proposed a method that allows a user to adjust the position of the mapping. These methods refer to the texture from the object information, source, and eye but do not assume light scattering across the scene. Therefore, they do not express colour bleeding and soft shadows. ACP and FTV convert the radiance to texture values by computing the global illumination and use the converted colour to compute the shading on the other object, so the texture values look as if it is reflected in the surroundings.

In stylisation, the shading of an object can be stylised from the rendered result using image processing. Magdics et al. [MSGS13] quantised the rendered pixel values to simplify the shading and emphasised it by shadow recolouring. Image analogies [HJO*01] transform the texture of an image using an example-based approach that transfers local features. StyLit [FJL*16] renders a 3D scene and stylises it using a style exemplar image provided by a user. It maps patches of the style image with the rendered image using the correspondence of the lighting conditions such as shadowing or scattering types. However, it does not transport the stylised colour in the scene. To represent colour bleeding and other lighting phenomena with such methods, the input drawing must depict these phenomena. ACP and FTV have a different purpose in that they compute the propagation of the light of a stylised object colour in the surroundings.

2.2. Rendering Algorithm

Kajiya formulated a photorealistic rendering algorithm as a rendering equation [Kaj86] and proposed a solution called path tracing (PT). PT traces all the light transport paths from each pixel into the scene. ACP converts the radiance estimated by PT into texture values at every scattering point on the remapping object to stylise its appearance.

PPM [HOJ08] is an unbiased estimator of the rendering equation that incrementally integrates the radiance from multiple photon maps. This process allows accurate estimation if a large number of photons is used. FTV converts the colour of the object radiance computed by PPM for stylisation.

2.3. Artistic Illumination Editing

We describe approaches that manipulate the scene parameters or light paths to produce the desired result while remaining in a physically based framework.

Ritschel et al. [ROTS09] proposed a system to modify mirror reflection. In this system, users can specify constraints to achieve artistic reflections. For global illumination, the work of Schmidt et al. [SNM*13] and Günther [GRR*16] enable caustic patterns to be manipulated by grouping light transport paths and moving the corresponding vertices. Kol et al. [KKSE16] proposed stylisation for light scattering in participating media. This approach enables the design of light shafts using fake occluders, and users can specify the colour of the light shafts. Unlike methods that control light transport paths geometrically, our methods replace the physically based radiance with a specified colour gradation without modifying the paths.

It is helpful to provide artists with an intuitive system for designing material parameters so that the rendered result matches the desired surface appearance. For such a goal-based workflow, methods defining consistent parameter sets for BRDFs [GGT*20] and providing an interface to adjust the parameters [NSRS13] have been proposed. However, these methods do not consider global illumination with arbitrary colours defined by users, as our methods do.

3. Preliminaries

ACP and FTV are based on a 3D computer graphics rendering that outputs a 2D image from a scene described in a 3D coordinate space. We refer to the object in the scene to be colour converted as the remapping object. We map a colour to another colour to edit the appearance of the remapping object with a 1D texture. In this paper, the colour is represented by three values, red (R), green (G), and blue (B). Therefore, the texture-colour remapping is $\mathbb{R}^3 \rightarrow \mathbb{R}^3$.

4. Method 1: ACP

We convert the radiance on the remapping object to the texture values. This enables a user to stylise the shading by specifying its colour with a 1D texture. In addition, it reproduces the colour bleeding of the texture values on the surrounding objects (Figure 2). In ACP, if a path travels through multiple objects, the remapped colour at the previous intersection affects the next intersection.

ACP is based on PT. For clarity of explanation, here we assume that the PT samples a single path to estimate the radiance and integrates the contribution over all sample paths. On the remapping object, we obtain the texture colour at the value u, converted from the estimated radiance $\langle L_p \rangle$ by sampling one incoming direction. We use the colour on the texture as the colourising factor of each path. We also calculate the average radiance of the RGB channels as the weight of the texture. We convert the estimated radiance into the XYZ colour space and use its Y coordinate y. The weight w is y but with minimum weight w_{min} to prevent the resulting value from appearing too dark.

$$w = max(y, w_{min}) \tag{1}$$

The value $u \in [0,1]$ used to refer to the texture values is defined as the relative position of y in an interval $[y_{min}, y_{max}]$, which the user specifies.

$$u = \begin{cases} 1 & \text{if } y_{max} \leq y \\ (y - y_{min})/(y_{max} - y_{min}) & \text{if } y_{min} \leq y \leq y_{max} \\ 0 & \text{if } y \leq y_{min} \end{cases} \tag{2}$$

ACP computes the non-photorealistic radiance using the following steps:

1. Determine the first intersection of the scene with the path traced from a point on a pixel.
2. Sample one incoming direction to estimate the radiance.
3. If the intersection is on the remapping object, perform steps 3a and 3b.

   3a. Convert the radiance to texture values and calculate w.
   3b. Multiply the texture values by w to obtain the radiance.

4.1. Stylised Shading Based on PT

Our aim is to derive the non-photorealistic radiance in direction $\omega \in \mathcal{S}^2$ at a point x on the remapping object from the following rendering equation [Kaj86]:

$$L_p(x, \omega_o) = L_e(x, \omega_o) + \int_{\mathcal{S}^2} f(x, \omega_i, \omega_o) L_i(x, \omega_i) |n(x) \cdot \omega_i| d\sigma(\omega_i) \tag{3}$$

where $\mathcal{S}^2$ is the unit sphere on $\mathbb{R}^3$, $n(x) \in \mathcal{S}^2$ is the surface normal at x, $\omega_i \in \mathcal{S}^2$ is the incoming direction, $L_e(x, \omega_o)$ and $L_i(x, \omega_i) \in \mathbb{R}^3$ respectively are the emission and the incidence as colour, and $f : (x, \omega_i, \omega_o) \rightarrow \mathbb{R}$ is a bidirectional scattering distribution function (BSDF). The Monte-Carlo estimator draws one sample direction ω_i from a proposal distribution p to obtain

$$\langle L_p \rangle = L_e(x, \omega_o) + \frac{f(x, \omega_i, \omega_o) L_i(x, \omega_i) |n(x) \cdot \omega_i|}{p(\omega_i)} \tag{4}$$

Here, we convert $\langle L_p \rangle$ to a value $u \in [0,1]$, as described above, and use it to refer to 1D texture t. With the averaged $\langle L_p \rangle$ and monochrome radiance w, we amplify colour t to obtain the remapped radiance for each path. Finally, we integrate the remapped radiance of all sample paths, which is expressed as

$$L_{NP} = \int_{\mathcal{S}^2} wt(u) p(\omega_i) d\sigma(\omega_i) \tag{5}$$

Note that texture t is not the radiance but the base colour of an object, and parameter w is the averaged radiance for the colour channels. Therefore, wt is the radiance. Because we use one sample estimation $\langle L_p \rangle$ to obtain w and u to stylise the radiance, texture colour $t(u)$ also depends on p.

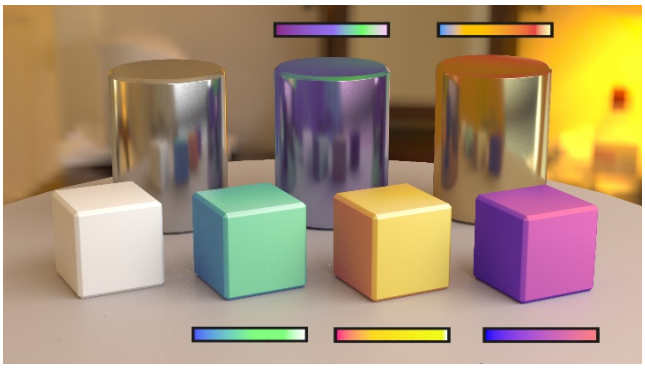

Figure 2: *Comparison of PT (leftmost objects) and ACP with 1D textures applied (other objects).*

4.2. Result

We implemented ACP in a path tracer that combines BSDF oriented sampling and explicit light sampling. We compared ACP with extended implementations of cel shading [LMHB00] and colour remapping for 2D images [MSGS13]. For the former method, we implemented A1, A2, and A3, which map 1D texture onto the angle between the normal and incident direction. A1 computes $u = max(n(x) \cdot \omega_i, 0)$ with an ω_i sampled in the direction toward the light sources. A2 maps a texture based on the direct illumination. We then used Equation 2 with I as the direct illumination to refer to the texture values. A3 considers indirect illumination. The material

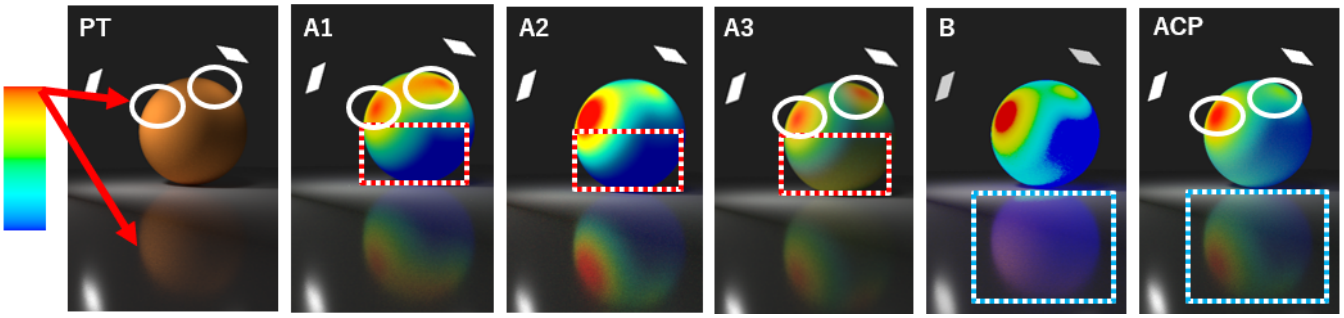

Figure 3: *Comparison of global illumination reproduction methods, showing the effect of indirect incidence on the texture values (red and white dashed rectangles), reflection (blue and white rectangles), and dependency of texture values on the object intensity (white circles).*

colour is determined by the texture values at $u = max(n(x) \cdot \omega_i, 0)$, with ω_i sampled by the path tracer, and the object is lit by the sampled incidence.

Image processing implementation B uses two inputs: the result of PT and a weight image that contains direct diffuse and inter-reflection components of the sphere. According to the weight, we blend the PT with the texture-converted image of PT.

When generating the results of each implementation (Figure 3), we used a scene containing a sphere with ideal diffusion, a floor with a diffuse and glossy component, and two area lights that illuminate the sphere. We applied a texture that returns a different hue to visualise the map from the intensity to texture values (Figure 3 left).

The results of methods A1 and A3 in Figure 3 show that the blue in the lower part of the texture, appears in the red and white boxed areas because these methods do not consider the reflection from the floor to the sphere. In the areas indicated by white circles, red appears (the upper part of the texture) because the results of A1 and A3 in Figure 3 depend only on the incoming direction whereas PT yields a different intensity. In addition, the results of A3 in Figure 3 present a photorealistic shading by path-traced illumination. However, the darkest part is not blue. This problem occurs because the texture coordinates are computed independently of the shading.

The results of implementation B shown in Figure 3 yield colours that incorporate the reflection from the floor of the sphere. However, the floor does not correctly reflect the texture values on the sphere. In the result obtained by ACP, the colour corresponds with the indirect light from the floor and the intensity of the shading. Furthermore, texture values are incorporated in the reflection in the floor.

4.3. Limitations and Discussion of ACP

Figure 4 shows two results rendered with conventional PT and with our colour conversion approach. To test our method, one cylinder, two cubes, and a bumpy sphere were used as remapping objects. Because there is a difference in calculation time depending on the brightness of the resulting image, we aligned the conditions by inputting a pure white texture to generate almost the same output as that of PT. It is important in this test for the two output images

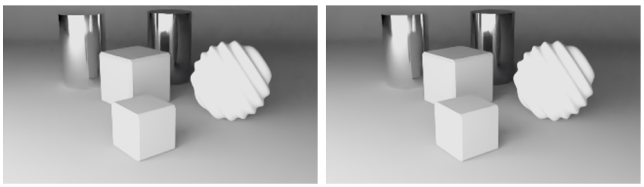

Figure 4: *Scene for the computational time comparison.*

to be almost the same. The image dimensions are 960 × 540, and we sampled 1000 paths per pixel. The PT rendering took 38.67 s, and our method took 37.04 s. We performed both processes with an AMD Ryzen 9 3950X CPU. Both processes took almost the same time because the colour conversion of ACP is performed in the material evaluation of the PT and only needs to sample one incoming path, as does the conventional PT algorithm.

ACP yields different results depending on the renderer implementation because we apply a nonlinear texture transform to the estimated radiance of each sample path, which depends on the renderer. However, ACP could be a user's preferred method because of its validity when focusing on a single path and providing a smooth interpolation result for the input colour. The dependency on the renderer is solved by averaging the samples in FTV.

We proposed a non-photorealistic rendering algorithm for PT. It converts radiance to texture values and reproduce reflections in the surroundings better than existing methods. Although we do not show the results, we confirmed that ACP also reproduces refraction and caustic patterns. It is an effective approach to consider the effect of shading separately from the texture values. However, the variance in the sampler and the nonlinear colour conversion during importance sampling cause mixing in the assigned texture values and damage the result. If a quantised texture is applied, ACP yields blurred boundaries instead of clear ones (Figure 5). Therefore, improvements are needed to exactly represent the colour in the input texture. In addition, storing the radiance values that have already computed enables textures to be edited after rendering. We hence propose FTV as a solution to this issue.

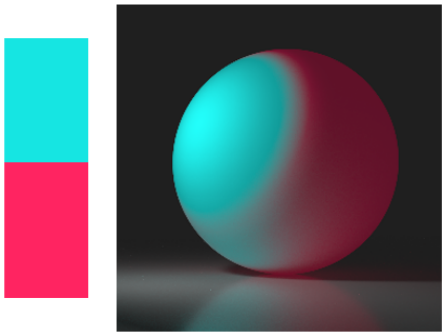

Figure 5: *Texture with two colours (left) applied using ACP (right).*

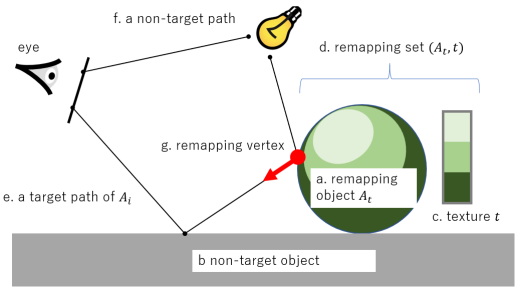

Figure 6: *Overview of FTV. We apply the colour conversion to the vertices of the target paths closest to the eye.*

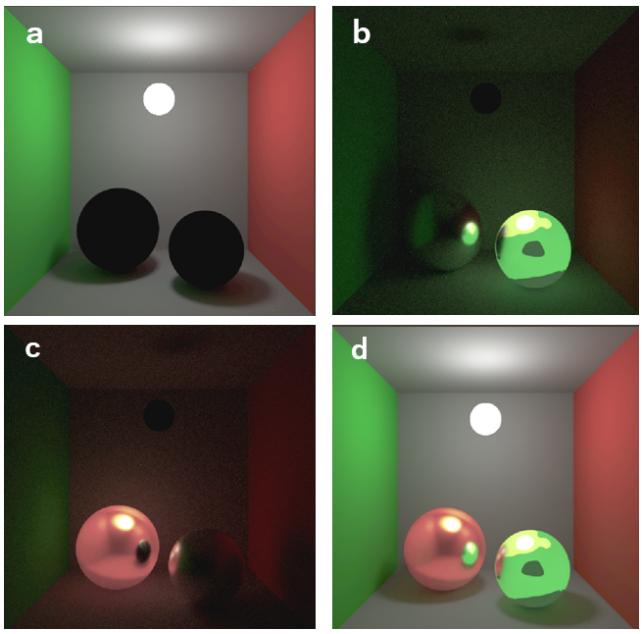

Figure 7: *Composition of FTV. The components of the non-target objects (a); the target paths of the red or green (b, c) spheres; and the sum of each component (d).*

5. Method 2: FTV

5.1. Overview

We define the target path as a light transport path that has vertices on a remapping object (Figure 6a). An object that is not the remapping object is defined as a non-target object (Figure 6b), and a path in which all vertices are on non-target objects is defined as a non-target path (Figure 6f). We assign a texture t (Figure 6c) to each remapping object, and we define the assignment (A_t, t) the remapping set (Figure 6d). The vertex of a target path where we convert the radiance is defined as the remapping vertex (Figure 6g).

FTV calculates the components of all non-target paths (Figure 7a) and components of the target paths of each remapping object (Figure 7b, c) that is chosen without overlap from the union of all the target paths of all remapping objects. In Figure 7, the two spheres reflect light from each other to yield a path that belongs to the target path of both the spheres, but such a path is classified as either a contribution component of b or c in Figure 7. In our implementation, the classification is based on the material of the remapping object. If the remapping object is diffuse, we select the closest intersection point from all vertices on each single path that goes from the remapping object to the eye. For a specular remapping object (which is not diffuse), we do not store its intersection points but instead store the points on the object in the reflection to produce the reflecting colour on the resulting image.

To obtain the components for each remapping object, we compute the radiance of the remapping vertices and accumulate the arbitrarily converted colour with the assigned texture from the radiance. Finally, we total all components (Figure 7a, b, and c) to obtain the output image (Figure 7d).

5.2. Computation of the Contribution of All the Target Paths of a Remapping Object

For clarity of explanation, here we assume that PPM collects a large sample for each intersection point using photon maps for consistent estimation. For each remapping object, similarly to PPM, we convert its radiance using the following three steps.

1. Ray tracing phase: We trace paths from the eye in the scene and store one remapping vertex on the remapping object for each path (Figure 6g, also indicated by boxes in Figure 8). The attenuation rates of each R, G, and B channel through the subpath are stored along with each remapping vertex. The position of a remapping vertex should belong to remapping objects. Each attenuation rate of the R, G, and B channels belongs to the range [0,1]. To generate the results in this paper, we classified the paths in the ray tracing phase so that diffuse reflective objects return the texture values assigned to themselves and objects with specular reflective components appearing to be the colour of the reflected object.
2. Photon tracing phase(s): After phase 1, we create a photon map in which each photon travels along arbitrary path starting from the light source (the dots in Figure 8). We use this map to estimate the radiance on the remapping vertices (the boxes in Figure 8) that were stored in the first phase. The estimation belongs

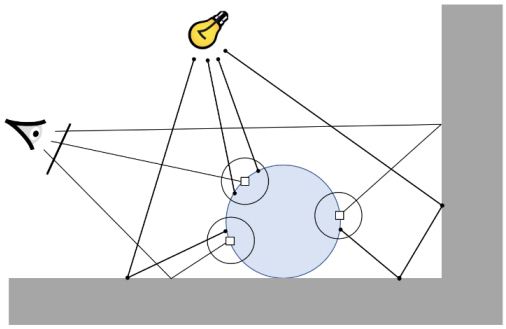

Figure 8: *Radiance estimation on a remapping object. First, we trace rays to determine and store remapping vertices (boxes). A photon tracing phase creates a photon map (dots) to calculate the radiance on each remapping vertex.*

to a 3D positive real coordinate space and is stored with each remapping vertex.

3. Accumulation: For each remapping vertex, we convert the radiance using the assigned texture, and the multiplication of the texture values by the attenuation rates are added to the corresponding pixel.

This ray tracing phase differs from common photon mapping. Whereas photon mapping stores intersection points on the first diffuse object during ray tracing, the proposed method continues ray tracing until it reaches the remapping object. For a path that travels over multiple remapping objects, we must choose one of them. In Section 5.3, we describe how we classified those paths to generate the results in this paper.

We use PT to obtain the contribution of all non-target paths (Figure 7a) separately from those of the target paths. Finally, we sum all components (Figure 7a, b, c) to obtain the output image (Figure 7d).

5.3. Implementation

We use PT to obtain the contribution of all non-target paths. We then use PPM to obtain the radiance of the remapping object.

So that diffuse reflective objects return the texture colour assigned to themselves and objects with specular reflective components appear to be the colour of the reflected object, we use the following procedure to determine the remapping vertex for estimating the photon density.

Determine the intersection x of the ray tracing that is closest to the eye and choose one of the following three cases.

Case 1. If x is a diffuse remapping object, store it as the remapping vertex

Case 2. If x is a non-target object, randomly choose the scattering direction in which to continue ray tracing and continue the process according to one of the three cases.

Case 3. If x is a specular remapping object, trace the reflected ray to determine the next intersection x' and proceed to step a, b, or c.

a. If x' is a remapping object and has a non-specular component, store it as the remapping vertex.
b. If x' is a remapping object and has specular component, trace the reflected ray at x' to determine next intersection x''. Substitute x" for x', and then proceed to step a, b, or c.
c. If x' is a non-target object, store x as the remapping vertex.

Here, we assume only perfectly diffuse or perfectly specular materials. A mixed material could be possible if we implement a stochastic decision for Cases 1 or 3 according to the ratio of diffusion and specularity.

We use a 1D texture, $[0,1] \rightarrow \mathbb{R}^3$, which maps a value to a colour. We convert the estimated radiance to the texture values.

5.4. Result

We compared the effect of global illumination of FTV with existing methods. The existing methods can be divided into two groups: those that use the scene geometry and those that use the rendered pixel values. For the first group, we chose cel shading [LMHB00] as a comparison method. For the second group, we implemented an image processing method that considers the interreflections among the surroundings. The cel shading method uses the texture values from the dot product of the direction to the light source and the normal. The scene used in this comparison has a sphere light, but we treat it as a point light at its centre for this vector math because cel shading does not deal with surface lights such as sphere lights.

In the image processing method, we replaced the pixel values of a PT image (Figure 9, PT) with a texture. To determine which pixels to use for each colour conversion, we prepared weight images (Figure 9, fr and fl), which are the sum of the direct reflection and interreflection components of each remapping object. We present the result of each colour remapping with the texture in Figure 9, cr and cl. When multiple remapping objects reflect each other, we choose pixels that have the higher reflection count from cr or cl to ensure each sphere represents the colour conversion of the other.

We compared the existing methods and ours using two scenes that include caustics (Section 5.4.1) and refraction (Section 5.4.2), respectively. In both comparisons, our results retained the shape and colour characteristics of shading. In Section 5.4.3, we compare the result of each method using a practical box scene. In Section 5.4.4, we reproduce an existing expression style with FTV.

5.4.1. Comparison of Caustics

We compared a scene in which a caustic pattern is created on the remapping sphere (Figure 10) by light through a glass sphere between the remapping object and the light source. As a result, the texture values of the cel shading do not correspond with those of the caustic pattern. This is because the result of the remapping object only depends on the position of the light source, not the glass sphere. The texture values of the image processing and FTV results correspond with the correct shading on the sphere.

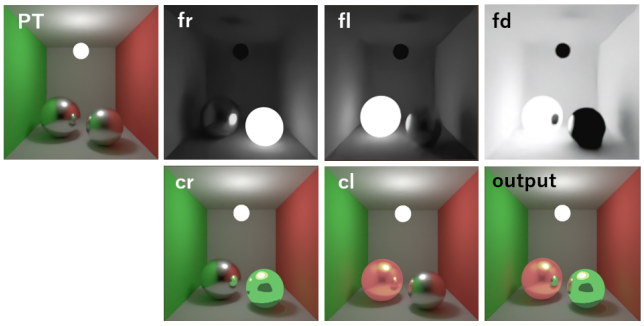

Figure 9: *Image processing method compared with FTV: (PT) result of path tracing; (fr, fl) ratio of direct or indirect reflection of each sphere through the pixel values; (cr, cl) weighted sum of PT and the converted colour with fr and fl as the weights; (fd) weights used to mix cr and cl, where the colours of cr and cl appear as blacker or whiter regions, respectively; (output) each channel of each pixel in the image is computed using cl*fd + cr*(1 - fd).*

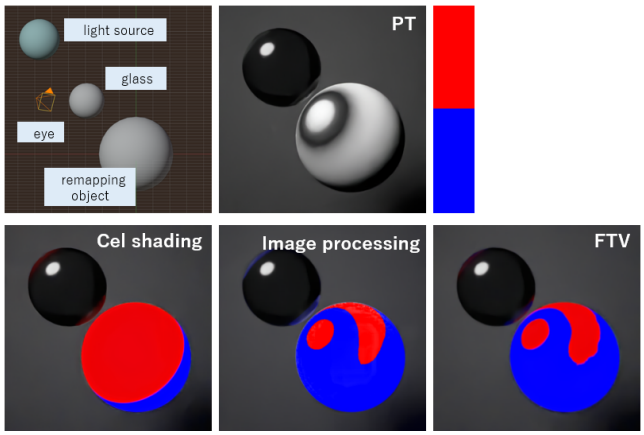

Figure 10: *Comparison of caustics: (top) the entire scene, the result of PT, and the applied texture; (bottom) the results of each method.*

5.4.2. Comparison of Refraction or Transmission

We compared the results for situations in which the remapping object is refracted in the surroundings. We used a scene in which the camera looks the remapping sphere through a glass object (Figure 11). The image processing result yielded colours other than those in the input texture in the refraction because it refers to the colour of the pixels, which is attenuated by transmission through glass.

5.4.3. Mutual Reflection Between Remapping Objects

We rendered a scene that contains two glossy spheres (Figure 12). The result of cel shading depends on the view direction and the position of the light source. Therefore, it did not convert the shading considering the surrounding spheres or walls, which the image processing method and FTV do. In addition, cel shading yielded only values that exist on the assigned texture, whereas image processing

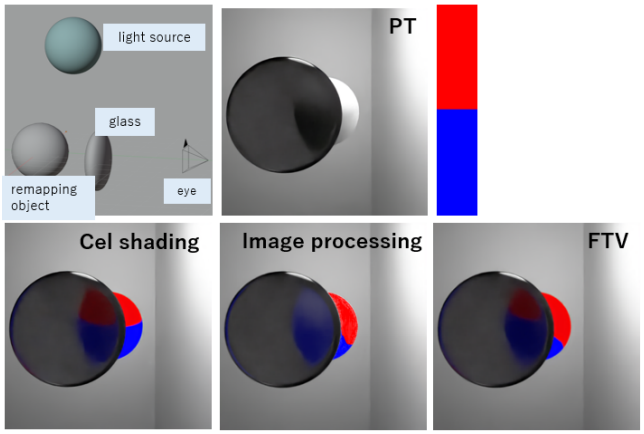

Figure 11: *Comparison of refraction: (top) the entire scene, the result of PT, and the texture we applied; (bottom) the results of each method.*

and FTV methods reflect the colours of the other sphere (Figure 12, yellow boxes), cell shading only shows the texture values assigned to each object.

In the image processing result, the colour of the object affects the surroundings in the same way as in FTV (mainly in the shadows). Whereas FTV requires the estimation of the radiance of each remapping object, the image processing approach requires a single rendering result and a weight image for each object, therefore the computational cost is small.

5.4.4. Other Results

We produced an existing style of expression with FTV. In the reproduction of the stylised shading in Figure 13, we were able to express the reflection on the water surface (the yellow reflections) and the colour change along the soft shadow (the red shadows).

5.5. Rendering Time

We profiled the computational time required to render the scenes of our results. We performed all process on an Intel Core i7 8700 processor. The parameters for PPM and rendering time of each phase are shown in Table 1. In our implementation, we performed all three phases for each remapping object. We discuss the computational cost with respect to ACP in Section 6.

5.6. Limitations and Discussion of FTV

As explained in Section 5.3, a target path that travels to remapping objects is classified to one remapping in the ray tracing phase. If we select target paths more flexibly, it may be possible to express more complex scenes. In the scene shown in Figure 12, the red sphere, which is reflected further into the reflected green sphere, is not represented, but this issue may be addressed by selecting the target path well. Another issue is that once a remapping set has been determined, it cannot be changed in later phases.

In both proposed methods, the BSDF of the remapping object

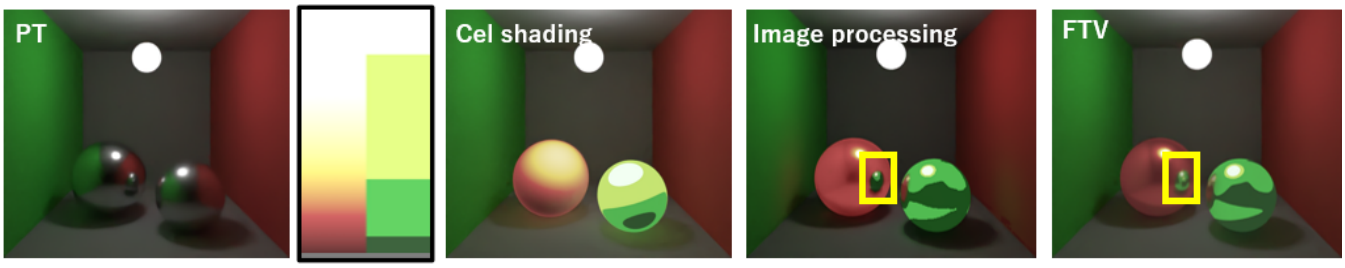

Figure 12: *Comparison of global illumination effects. Image processing and FTV reproduce mutual reflections of the two spheres (indicated by yellow boxes). The textures are inputs for the cel shading and our method.*

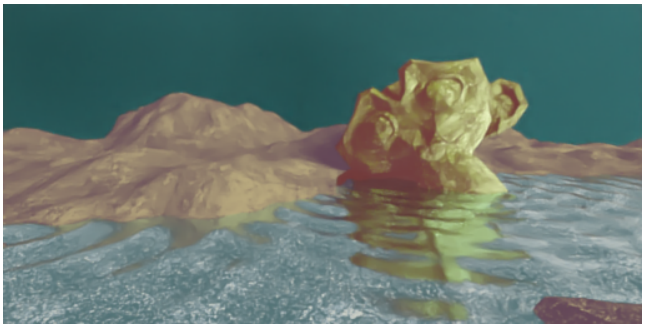

Figure 13: *Reproduction of stylised shading.*

cannot be replaced after the radiance has been estimated. If it is possible, we can more flexibly edit the object appearance after rendering. To create the appearance of different roughness and refractive indices of an object, it is necessary to render again after changing the material parameters.

In addition, importance sampling on paths with large contributions is not always effective for FTV, unlike in general path-finding methods. For example, when applying a texture that inverts the colour, we should focus on paths with a low contribution to the photorealistic radiance (and with a large contribution to the final result) to enable the convergence of the pixel values. In particular, PPM is less likely to improve the quality of dark areas because the convergence of the estimated values is faster in areas with more photons.

Although the implementation of FTV uses PPM to estimate the radiance of the remapping objects, other estimators can be used. Whereas general photon mapping including PPM collects photons only on diffuse objects, FTV places the intersection points on remapping objects regardless of its material. Therefore, when a material with a sparse BSDF, such as a shiny material, is applied to the remapping object, the PPM radiance estimation becomes inefficient. This is because photon mapping does not focus on the direction of incidence to the object. In this case, it is better to choose a sampling method that considers the BSDF using PT to improve the results. Alternatively, as done in general photon mapping methods, we can apply PPM by extending the path from the eye until it reaches a diffuse object, instead of estimating the radiance on glossy objects.

6. Discussion

We discuss the computational costs of ACP and FTV as well as the difference between them. Our two methods are suitable for different situations. ACP is reasonable when the aim is to focus on remapping every single path we trace, whereas FTV is reasonable when focusing on remapping the outgoing light from the surface. In addition, users can choose ACP to shift the input colours smoothly. FTV, by contrast, uses only the colour of the input texture for the output.

The colour conversion of ACP is performed in the material evaluation of the PT and requires one incoming path to be sampled. Thus, the order of this method is the same as that of conventional PT, as shown in Section 4.3. By contrast, FTV estimates the radiance for each intersection point, and the overall amount of computation is proportional to the size of the samples in each of the two phases to obtain the pixel vales. To improve the quality of the output, both the sample size of the paths connecting the eye and the remapping object for each pixel as well as the sample size for radiance estimation must be sufficiently large. Hence, the amount of computation required is the product of the number of paths to be further branched by the number of paths in ordinary PT.

With multiple remapping objects, ACP converts the colour at all vertices on all remapping objects, whereas FTV selects one remapping object, avoiding duplication. The object radiance converted by ACP is used for later colour conversion on subsequent remapping objects. The final pixel values are the texture values assigned to the remapping object closest to the eye. It converts the brightness of the previously converted radiance with another texture. Conversely, in FTV, converted colours on the selected remapping object do not affect other colour conversions.

In a scene with complex inter-reflections, FTV requires one remapping object with texture to be chosen per path. However, we did not provide a user interface. We believe that visualising effective light paths [RKRD12] and selection based on path expression [SNM*13] could help users specify target paths in a complex scene. Another possibility for working with complex scenes is to incorporate the colour addition of multiple remapping objects partially or selectively, as in ACP. Although the aim of our research

is not to develop user interfaces, the implementations of our two methods will allow users to edit textures and check the results interactively. Hence, the user can easily obtain the final result through trial and error.

7. Conclusion

We proposed two methods, ACP and FTV, which enable stylised shading of 3D scene to be rendered considering the global illumination. We showed that our methods determine the texture values reproducing the effects of global illumination with other objects when compared with existing methods.

ACP stylises shading using PT. The converted texture values affect the remapping in the next remapping on a PT path. Moreover, it is effective to consider the effect of shading separately from the texture values. The resulting texture values are mixed because of the variance of the estimator.

Conversely, FTV provides the remapping result with only the colours of the specified input texture. Furthermore, this enables the radiance of the objects radiance to be converted afterward. In the future, we would like to perform multiple colour conversions with multiple remapping objects to create other artistic effects.

References

[AWB06] ANJYO K.-I., WEMLER S., BAXTER W.: Tweakable light and shade for cartoon animation. In *Proceedings of the 4th International Symposium on Non-Photorealistic Animation and Rendering* (New York, NY, USA, 2006), NPAR '06, Association for Computing Machinery, p. 133–139. URL: https://doi.org/10.1145/1124728.1124750, doi:10.1145/1124728.1124750. 2

[Ble] Shader to rgb - blender 2.92 reference manual [online]. Last checked: 2021-04-26. URL: https://docs.blender.org/manual/en/latest/render/shader_nodes/converter/shader_to_rgb.html. 1

[BTM06] BARLA P., THOLLOT J., MARKOSIAN L.: X-toon: An extended toon shader. In *Proceedings of the 4th International Symposium on Non-Photorealistic Animation and Rendering* (New York, NY, USA, 2006), NPAR '06, Association for Computing Machinery, p. 127–132. URL: https://doi.org/10.1145/1124728.1124749, doi:10.1145/1124728.1124749. 2

[FJL*16] FIŠER J., JAMRIŠKA O., LUKÁČ M., SHECHTMAN E., ASENTE P., LU J., SÝKORA D.: Stylit: Illumination-guided example-based stylization of 3d renderings. *ACM Trans. Graph. 35*, 4 (July 2016). URL: https://doi.org/10.1145/2897824.2925948, doi:10.1145/2897824.2925948. 1, 2

[GGT*20] GUARNERA D., GUARNERA G. C., TOSCANI M., GLENCROSS M., LI B., HARDEBERG J. Y., GEGENFURTNER K. R.: Perceptually validated cross-renderer analytical brdf parameter remapping. *IEEE Transactions on Visualization and Computer Graphics 26*, 6 (2020), 2258–2272. doi:10.1109/TVCG.2018.2886877. 1, 2

[GRR*16] GÜNTHER T., ROHMER K., ROESSL C., GROSCH T., THEISEL H.: Stylized caustics: Progressive rendering of animated caustics. *Computer Graphics Forum 35* (05 2016), 243–252. doi:10.1111/cgf.12827. 1, 2

[HJO*01] HERTZMANN A., JACOBS C. E., OLIVER N., CURLESS B., SALESIN D. H.: Image analogies. In *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques* (New York, NY, USA, 2001), SIGGRAPH '01, Association for Computing Machinery, p. 327–340. URL: https://doi.org/10.1145/383259.383295, doi:10.1145/383259.383295. 2

[HOJ08] HACHISUKA T., OGAKI S., JENSEN H. W.: Progressive photon mapping. *ACM Trans. Graph. 27*, 5 (Dec. 2008). URL: https://doi.org/10.1145/1409060.1409083, doi:10.1145/1409060.1409083. 2

[Ich15] ICHIKAWA H.: *land of the lustrous*, vol. 4. Kodansha, 2015. 1

[Kaj86] KAJIYA J. T.: The rendering equation. In *Proceedings of the 13th Annual Conference on Computer Graphics and Interactive Techniques* (New York, NY, USA, 1986), SIGGRAPH '86, Association for Computing Machinery, p. 143–150. URL: https://doi.org/10.1145/15922.15902, doi:10.1145/15922.15902. 2, 3

[KKSE16] KOL T., KLEHM O., SEIDEL H.-P., EISEMANN E.: Expressive single scattering for light shaft stylization. *IEEE Transactions on Visualization and Computer Graphics 23* (04 2016), 1–1. doi:10.1109/TVCG.2016.2554114. 2

[LMHB00] LAKE A., MARSHALL C., HARRIS M., BLACKSTEIN M.: Stylized rendering techniques for scalable real-time 3d animation. In *Proceedings of the 1st International Symposium on Non-Photorealistic Animation and Rendering* (New York, NY, USA, 2000), NPAR '00, Association for Computing Machinery, p. 13–20. URL: https://doi.org/10.1145/340916.340918, doi:10.1145/340916.340918. 1, 2, 3, 6

[MSGS13] MAGDICS M., SAUVAGET C., GARCÍA R. J., SBERT M.: Post-processing npr effects for video games. In *Proceedings of the 12th ACM SIGGRAPH International Conference on Virtual-Reality Continuum and Its Applications in Industry* (New York, NY, USA, 2013), VRCAI '13, Association for Computing Machinery, p. 147–156. URL: https://doi.org/10.1145/2534329.2534348, doi:10.1145/2534329.2534348. 1, 2, 3

[NSRS13] NGUYEN C. H., SCHERZER D., RITSCHEL T., SEIDEL H.-P.: Material Editing in Complex Scenes by Surface Light Field Manipulation and Reflectance Optimization. *Computer Graphics Forum (Proc. EUROGRAPHICS 2013) 32*, 2 (2013). 1, 2

[RKRD12] REINER T., KAPLANYAN A., REINHARD M., DACHSBACHER C.: Selective inspection and interactive visualization of light transport in virtual scenes. *Computer Graphics Forum 31* (2012). 8

[ROTS09] RITSCHEL T., OKABE M., THORMÄHLEN T., SEIDEL H.-P.: Interactive reflection editing. In *ACM SIGGRAPH Asia 2009 Papers* (New York, NY, USA, 2009), SIGGRAPH Asia '09, Association for Computing Machinery. URL: https://doi.org/10.1145/1661412.1618475, doi:10.1145/1661412.1618475. 1, 2

[SMGG01] SLOAN P.-P. J., MARTIN W., GOOCH A., GOOCH B.: The lit sphere: A model for capturing npr shading from art. In *Proceedings of Graphics Interface 2001* (CAN, 2001), GI '01, Canadian Information Processing Society, p. 143–150. 1, 2

[SNM*13] SCHMIDT T.-W., NOVÁK J., MENG J., KAPLANYAN A. S., REINER T., NOWROUZEZAHRAI D., DACHSBACHER C.: Path-space manipulation of physically-based light transport. *ACM Trans. Graph. 32*, 4 (July 2013). URL: https://doi.org/10.1145/2461912.2461980, doi:10.1145/2461912.2461980. 1, 2, 8

[TAY13] TODO H., ANJYO K., YOKOYAMA S.: Lit-sphere extension for artistic rendering. *Vis. Comput. 29*, 6–8 (June 2013), 473–480. URL: https://doi.org/10.1007/s00371-013-0811-7, doi:10.1007/s00371-013-0811-7. 1, 2

Table 1: *Rendering parameters and time of our results*

	Figure 10	Figure 11	Figure 12	Figure 13
Image dimension	512*512	512*512	512*512	512*256
Ray per pixel	4	16	16	16
Photon emission per a photon tracing phase	20000	20000	1000	10000
Number of iterations	100	100	200	1000
Radius reduction ratio for photon collection	0.6	0.6	0.6	0.6
Initial radius to collect photons	1	1	1	1
Ray tracing phase	0.10 s	1.12 s	1.79 s	16.64 s
Photon tracing phases	15.64 s	11.46 s	103.82 s	3477.12 s
Accumulation	1.35 s	1.12 s	4.19 s	27.47 s