Multiple Importance Sampling

Collin Potts and Madelyn Torres

December 9, 2019

Abstract

This paper discusses an implementation of the bidirectional path tracing algorithm (BDPT) using multiple importance sampling (MIS). We present a Monte Carlo method for estimating the overall illumination of each pixel. We also discuss the types of light sources used in our implementation as well as how rays are sampled from them. We further address the effects of varying key parameters such as the expected length of generated paths and the number of samples we generate for each pixel. Lastly, we discuss how paths are weighted using MIS.

Introduction

The task of simulating light in a scene is a fundamental step in most computer graphics rendering pipelines. In the physical world, lighting is observed when photons are emitted from a source and, after continuous interactions with the environment, collide with a viewer's eye. The process of emulating this process is known as global illumination. Global illumination is computationally difficult to capture accurately. As a result, there is often a trade-off between lighting accuracy in a scene and computational efficiency.

To overcome the drawbacks of performing this trade-off, extensive research has been performed in lighting algorithms which are both accurate and relatively inexpensive computationally. One such class of algorithms is known as light transport. In light transport algorithms, light paths are generated from a light source and their interactions with a scene are calculated before determining their visibility to the viewer. BDPT is a type of light transport algorithm which generates paths of light from a light source for physical accuracy and combines them with paths generated from the viewer for efficiency purposes.

The goal of this paper is to demonstrate that BDPT is an effective method for not only generating more realistic scenes compared to more naive ray tracing, but also that it does so in an efficient manner. These benefits are further demonstrated across a variety of sampling methods, types of light sources, and parameters for sampling. Lastly, we demonstrate that BDPT can be improved even further by using MIS to weight the contribution of sampled paths to the overall illumination of a pixel.

Background

Our work is based largely on the overview of Bidirectional Path Tracing which composes a chapter of Eric Veach's dissertation, a seminal work in Monte Carlo methods for Light Transport Simulation [Veach, 1997]. Bidirectional path tracing leverages the concept of generating two sub-paths: one which starts from the camera origin and another which is starts from a sampled light source. Following Veach's notations, we can refer to the light sub-path as a set of points $y_0, ..., y_{n_L-1}$, where n_L is the number of points in the light path. Identically, the eye sub-path can be notated as $z_0, ..., z_{n_E-1}$. The lengths of these sub-paths is determined randomly, as we discuss further in our implementation section. Given these two paths, we can define a sample path $\bar{x}_{s,t} = y_0, ..., y_{s-1}, z_{t-1}, ..., z_0$, tracing from the light source to the camera. The benefit of this path formulation is that we can easily identify paths for a ray of light to reach the camera, which would be difficult with a unidirectional approach. Veach provides a useful diagram showing how two paths may be joined to create a useful transport path, shown below in Figure 1.

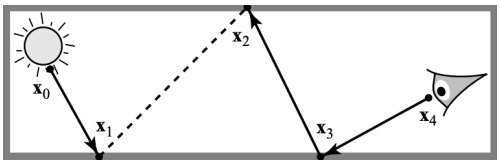


Figure 1: *An example of a BDPT transport path [Veach, 1997]*

Given our two sampled sub-paths, we can generate

the net light contribution as

$$L = \sum_{s=0}^{n_L-1} \sum_{t=0}^{n_E-1} w(\bar{x}_{s,t}) \frac{f(\bar{x}_{s,t})}{p(\bar{x}_{s,t})} \tag{1}$$

where w is our weighting function, p is our probability density function over paths, and f is our contribution function which measures how much light travels along the given path. p is determined according to the sampling method for path scattering, used to trace our sub-paths. f is measured according the amount of light being emitted and the BRDF values along our sub-paths. In order to calculate the $f(\bar{x}_{s,t})/p(\bar{x}_{s,t})$ value, we adapt the following equation from Vlnas's experimental survey of bidirectional path tracing [Vlnas, 2018]:

$$\begin{aligned} \frac{f(\bar{x}_{s,t})}{p(\bar{x}_{s,t})} = {} & L(\overrightarrow{y_0 y_1}) f_r(\overrightarrow{y_{s-2} y_{s-1}}, \overrightarrow{y_{s-1} z_{t-1}}) \\ & f_r(\overrightarrow{y_{s-1} z_{t-1}}, \overrightarrow{z_{t-1} z_{t-2}}) \\ & \prod_{i=1}^{s-2} \frac{f_r(\overrightarrow{y_{i-1} y_i}, \overrightarrow{y_i y_{i+1}})}{p(\overrightarrow{y_{i-1} y_i}, \overrightarrow{y_i y_{i+1}})} \\ & \prod_{i=1}^{t-2} \frac{f_r(\overrightarrow{z_{i-1} z_i}, \overrightarrow{z_i z_{i+1}})}{p(\overrightarrow{z_{i-1} z_i}, \overrightarrow{z_i z_{i+1}})} \end{aligned} \tag{2}$$

where $f_r(\overrightarrow{ab}, \overrightarrow{bc})$ is the BRDF at point b with incoming vector $\overrightarrow{ab}$ and outgoing vector $\overrightarrow{bc}$. Additionally, $p(\overrightarrow{ab}, \overrightarrow{bc})$ is the probability of sampling the scattering vector $\overrightarrow{bc}$ given incoming vector $\overrightarrow{ab}$. Effectively, this equation breaks up the light contribution into the initial emitted light L, BRDF contributions of the connection between sub-paths, and the BRDF contributions of both sub-paths. Note that when we connect the two sub-paths, if there is any obstructing object, the contribution is instead 0, as no light is transported along this path.

The final component to be determined is the weight function w. The fundamental constraint of the weight function is that every weight must be non-negative and that the sum of weights must be 1. Formally,

$$\sum_{s=0}^{n_L-1} \sum_{t=0}^{n_E-1} w(\bar{x}_{s,t}) = 1, \quad w \geq 0. \tag{3}$$

A naive weighting system may provide the same weights to every sample, such that $w(\bar{x}_{s,t}) = \frac{1}{n_L n_E}$. However, we explore implementing Multiple Importance Sampling, which requires that we provide dynamic weighting, which provides more weight to more important samples. A proxy to importance for a sample $\bar{x}_{s,t}$ is of course the probability density of the path $p(\bar{x}_{s,t})$. With this in mind, we utilize the power heuristic described by Veach in his dissertation's chapter on MIS [Veach, 1997]. Under this definition,

$$w(\bar{x}_{s,t}) = \frac{p(\bar{x}_{s,t})^\beta}{\sum_{s'=0}^{n_L-1} \sum_{t'=0}^{n_E-1} p(\bar{x}_{s',t'})^\beta} \tag{4}$$

where the exponent β can be selected according to empirical variance reduction. While Veach suggests a value of $\beta = 2$, we explore different values of the parameter in our implementation.

Implementation

The code for our project can be found at: https://github.com/collinsgit/metro-caster [Potts and Torres, 2019].

Bidirectional Path Tracing

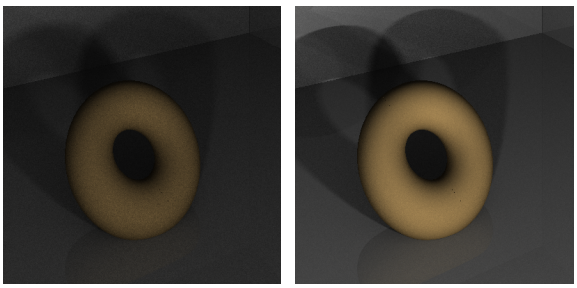
Below is pseudocode of our implementation of BDPT:

```
for pixel in scene:
    illumination = [0, 0, 0]
    for i in range(iters):
        eye_ray = gen_eye_ray(pixel)
        eye_path = gen_path(eye_ray,
            random(length))
        light_ray =
            sample_from_light(random(light))
        light_path = gen_path(light_ray,
            random(length))
        path = eye_path+connect(eye_path,
            light_path)+light_path
        weight = weigh(path)
        illumination += weight * trace(path)
    scene.set_pixel(illumination/iters)
```

In summary, we sample a user-defined number of paths for each pixel and average each sampled path's contribution to the overall illumination. In choosing a path, we generate a ray from the camera through the current pixel and sample a random direction from a random light source. Once we have both initial rays we trace paths of random lengths. Each hit along each path chooses a random new direction to continue in. Once the paths are generated, we connect them. The resulting path is then assigned a weight based on its probability of being sampled. The observed illumination is then adjusted based on this weight before being considered for the pixel's overall illumination.

It is important to note that when considering a sampled path we are in fact considering all pairs of possible sub-paths that can be made from connecting the eye path and light path. We ensure that each sub-path contains the initial ray cast from the eye to guarantee that the calculated illumination is visible. MIS (which is discussed in more detail later) is then applied to determine each sub-path's contribution to the path's overall illumination.

For our BRDF function f_r, as referenced in Equation 2, we utilize a standard Blinn-Phong implementation. We experiment with several types of scatter sam-



(a) Spherical Lighting (b) Area Lighting

Figure 2: *The two different types of light sources.*

pling procedures, resulting in different traced paths and probability density functions. These different sampling procedures are discussed in the **Samplers** section below.

Light Sources

Our implementation represents light in two ways:

- **Spherical Lighting**: Spherical lights are spheres which have a property that defines the intensity of emitted light. Light rays are sampled by selecting a point uniformly at random across the surface of the sphere and pointing it in the direction of the sphere's normal at that point.
- **Area Lighting**: Area lights are rectangles which also have a defined intensity of emitted light. Light rays are chosen by sampling an origin uniformly at random on the rectangle and point in the direction of the rectangle's normal.

If a light source is visible in a scene, its color is determined mostly by the intensity of its emitted light. The differences between the two types of lighting can be seen in Figure 2 where two light sources light an angled torus.

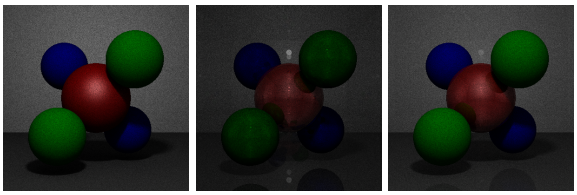
Samplers

All of our sampling procedures follow a simple format: input to the procedure is an incoming direction, here called θ_i and a material hit, which is simply a representation of the point on an object surface our ray is impacting. This provides us with the surface normal n at the point of impact, along with information about the specular (ρ_s) and diffuse (ρ_d) reflectances of the object material. The procedure provides an outgoing scatter direction θ_s. Knowing the sampling procedure, we can also calculate the probability distribution according to θ_i, θ_s, and our information about the surface. Below, we describe our 3 main sampling procedures.

- **Cosine Weighted Hemisphere:** This model produces the scatter direction θ_s with probability proportional to $\cos(\theta)$ where θ is the angle between the θ_s and the surface normal n. This model is especially useful for diffuse lighting, as the PDF is proportional to the diffuse contribution of incoming light. The exact PDF of the function is $(\theta_s \cdot n)/\pi$, which we can note to be independent of θ_i.
- **Pure Reflectance:** This procedure is quite simple, in that it always returns the scatter direction as $\theta_s = (\theta_i - 2*(\theta_i \cdot n)*n)$, which is then normalized. This direction maximizes specular contribution in the Blinn-Phong model. The PDF is then simply a constant value, as there is only one possible sampling. We note that this is an inaccurate model of specular reflectance. A more accurate model would be to perform rejection sampling to find a direction according to a cosine lobe around the specular direction, normalizing by the relevant spherical digon. We attempted this method, but found that it was difficult to implement and debug, and thus excluded it from our main results.
- **Hybrid Model:** Our final model is a hybrid of the preceding models. With some probability p, we sample according to the Cosine Weighted Hemisphere procedure. Otherwise, we sample according to the Pure Reflectance procedure. Aa a result, our sampling includes both diffuse and specular scattering, resulting in importance metrics which better align with the BRDF. To calculate the PDF, we simply take p times the Cosine Weighted Hemisphere PDF given the scattering direction plus $(1-p)$ times the PDF of Pure Reflectance. In order to best align our scattering PDF with the BRDF, we follow the guidance of the Global Illumination Compendium [Dutré, 2003] and specify

$$p = \frac{|\rho_d|_1}{|\rho_d|_1 + |\rho_s|_1}. \tag{5}$$

To better exemplify the 3 sampling procedures, we demonstrate the results of simulating the same scene with each procedure:



(a) Cosine Weighted Hemisphere (b) Pure Reflectance (c) Hybrid Model

Figure 3: *The three different sampling procedures.*

We note here that much of our implementation for these random sampling procedures derives from the equations and formulae provided in the Global Illumination Compendium, particularly for sampling from cosine weighted hemispheres, cosine lobes, and spherical digons [Dutré, 2003].

Multiple Importance Sampling

MIS is implemented as described in Equation 4. For each path that we sample, we begin with a total illumination value of 0. Every possible sub-path is then traced, each resulting in its own illumination value. These illumination values are multiplied with the cumulative PDF of its corresponding sub-path, i.e. the chance of that sub-path being sampled. This weighted illumination value is added to the total illumination. Lastly, the total illumination is divided by the sum of all sub-paths cumulative PDFs. It is important to note that each cumulative PDF is raised to the power of β. As mentioned earlier, this value is by default 2 but can be varied.

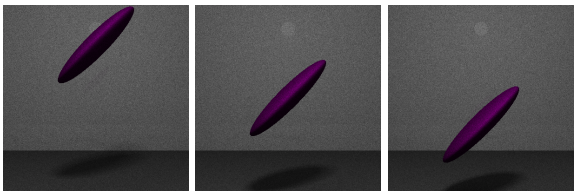
Results

Emergent Characteristics

In developing our BDPT system, equipped with MIS, we noted that in many of our simulated scenes we were able to observe emergent qualities of the images that were never specifically implemented. First, we noticed that an effect of providing no light transport across obstructed paths lead to the creation of realistic shadows in the image, with the sampling procedure resulting in fascinating examples of soft and hard shadows. Additionally, we observed that the pure reflectance portion of our sampling and lighting procedures lead to the creation of mirror-like reflections on specular materials.

Shadows

As mentioned, our process produces very realistic looking soft shadows. To better exemplify this, we simulate an ellipsoid in a box at different heights. With the light staying in the same place in the scene, we are able to observe the resulting shadow moving and becoming more crisp.



(a) High Height (b) Mid Height (c) Low Height

Figure 4: *An ellipsoid at different heights.*

Looking at these images, we can see that not only does the position of the shadow change as we lower the ellipsoid, but the shadow also becomes darker with more solid edges, matching the form of real shadows. This quality rises from the fact that we randomly sample positions on our light source when creating our light path, such that there are some points which will be in shade on one iteration but not on the next. As a result, we get a partial shading effect observed here.

Reflections

Highly specular surfaces are able to reflect incoming light, creating mirror-like effects. We can see an example of this reflection in the simulated image below of a torus.

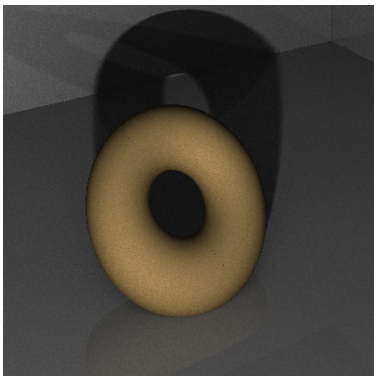


Figure 5: *A torus in a semi-reflective box.*

Not only can we see the torus reflected on the floor beneath, we can also see that the back wall reflects such that we can see the shadow of the torus in the reflection. Different configurations of material specularity and shininess can result in reflections varying from glossy to foggy in their appearance.

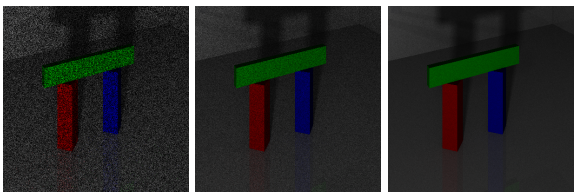
Parameter Effects

There are three tunable parameters in our implementation of BDPT:

- **Iterations**: The number of paths to sample for each pixel.
- **Length**: The length of each sub-path is sampled from a geometric distribution, parameterized such that each sub-path contains **length** points, in expectation.
- β**:** β is used as defined in Equation 4. Modifying β affects the level of contribution sub-paths have on the illumination of their overall path. Increasing β will reduce the variance of a sub-path's cumulative PDF.

Iterations

The number of iterations is positively correlated with the quality of an image. More specifically, increasing the number of iterations used for rendering an image will result in less noise. This is due to the fact that more samples are being taken to determine a pixel's resulting illumination. The effects of differing iteration numbers can be seen in Figure 6.

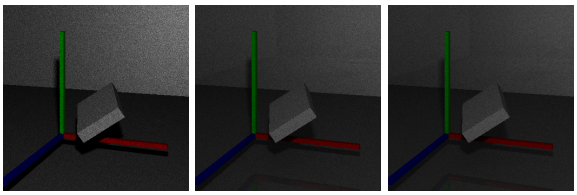


(a) 1 Iteration (b) 10 Iterations (c) 100 Iterations

Figure 6: *An arch made with differing amounts of iterations.*

Length

Though increasing the length increases the computation time, the effect is an image with higher quality reflections and less noise. This is naturally the result of sampling longer paths as more points in the scene are considered when estimating a pixel's illumination. The effects of varying length can be seen in Figure 7.



(a) Length 1 (b) Length 4 (c) Length 7

Figure 7: *Axes made using different lengths.*

β

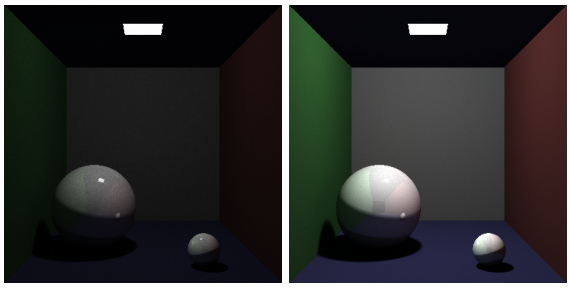
Though the effects of varying β are not as pronounced as varying length and the number of iterations, a visual difference is still perceptible. A larger β results in placing greater value on more probable paths. Consequentially, the impact of less probable paths is mitigated. The visual difference is that images appear a bit brighter, with specular lighting becoming more apparent. These effects can be observed in Figure 8.

(a) β =1 (b) β =2 (c) β =4

Figure 8: *Bunny rendered with different β.*

Evaluation Metrics

In order to more rigorously evaluate the quality improvements provided by MIS, we evaluate how quickly, in terms of iterations, our images converge to the simulation limit. By simulation limit, we mean the image that would be provided if we ran the simulation with arbitrarily many images, which we approximate by simply choosing a very large number of iterations. For both our BDPT with and without MIS, we create a reference "simulation limit" by running the approaches for 300 iterations. These are shown below:



(a) Without MIS (b) With MIS

Figure 9: *MIS and no-MIS reference images.*

As we can see, the two images look quite different, as MIS will weight shorter, brighter paths more, while the process without MIS will weight paths evenly. The major difference in these images supports our decision to use separate reference images. What remains to be seen is how quickly the two procedures converge to these reference images. To test this, we run our simulation with 1 to 100 iters, both with and without MIS. For every image generated, we evaluate the RMSE between that image and the corresponding reference. Note that pixel values are normalized to be in the range $[0, 1]$, rather than $[0, 255]$. Note the additional Figures 11 and 12, which exemplify the differences of adding MIS at several iteration values.

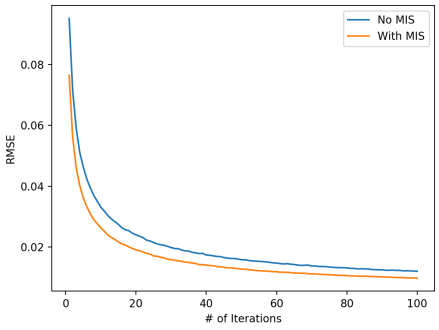
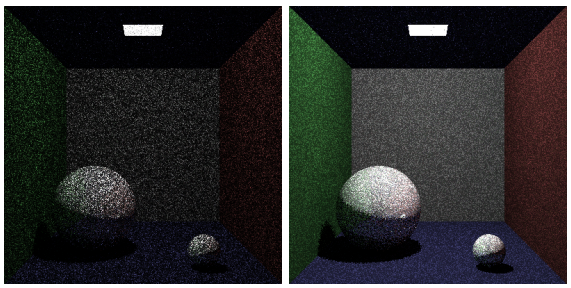


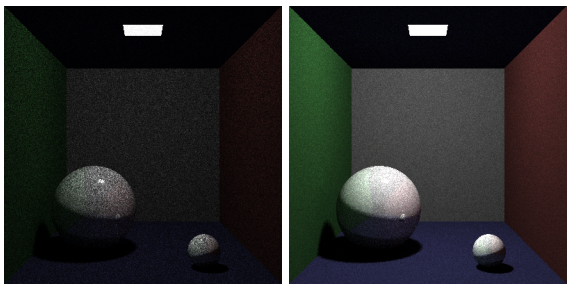
Figure 10: *RMSE between generated and reference images.*

We observe, as expected, that adding MIS allows the RMSE to start off lower and decrease more quickly. Note that this is in addition to the fact that MIS creates more physically realistic, light balanced images, as was demonstrated in Figure 9. Our only observed negative effect of utilizing MIS is that, by providing more weight to shorter paths, reflections were less visible. Perhaps



(a) Without MIS (b) With MIS

Figure 11: *MIS and no-MIS at 1 iteration.*



(a) Without MIS (b) With MIS

Figure 12: *MIS and no-MIS at 10 iterations.*

this could be improved by modifying the β parameter, as was lightly explored in the previous section.

Additionally, we note an incidental finding relating to file sizes. As described in Veach's original dissertation, MIS methods are meant to lower the variance of the resulting images, along with creating a more physically accurate image [Veach, 1997]. We may expected this to lead to images generated using MIS having smaller file sizes, as less variance may lead to easier compression. To explore this, we plot the filesizes of the images produced with and without MIS, for 1 to 100 iterations.

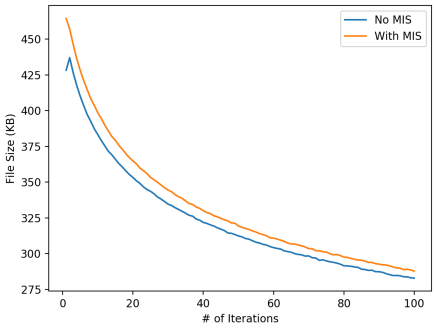


Figure 13: *File sizes of generated images.*

As we observe, MIS actually does not lower the file size, but performing more iterations can drastically reduce the resulting file size. To interpret this finding, we note that image compression is impacted less by variance of the image and moreso by the number of principal components needed to explain some percent of the net variance. Thus, while MIS may reduce the image variance, this does not imply that the number of necessary explanatory components decreases. However, more iterations reduces noise, which from a high level perspective may remove some higher-frequency signals, allowing for smaller compression.

Conclusion

In this paper, we have discussed an implementation of BDPT which uses MIS to weight randomly sampled eye-light paths. We have shown how our implementation of BDPT interacts with different types of light sources, as well as how its behavior varies with different tunable parameters, including path length, number of iterations, the MIS β parameter, and the samplers used to trace paths and calculate PDFs. Our resulting findings demonstrated that natural characteristics like shadows and reflections emerge from this system. Moreover, we explored the impact of MIS to demonstrate how it lowers image variance, ending with a finding on how the number of iterations can affect file size.

Future Work

Throughout our project, we considered many potential extensions to the project that we would consider to be natural continuations. These include:

- **Accurate Specular Scattering:** As mentioned previously, the specular scattering captured in our Pure Reflectance sampler is not physically accurate. A natural improvement would be to sample according to the cosine lobe centered around the specular direction (the reflection of the incoming direction across the surface normal). The resulting PDF would be proportional to $\cos^s(\alpha)$, where s is the shininess of the surface and α is the angle between the outgoing direction and the specular direction.
- **MIS Across Iterations:** Our current model utilizes MIS on all of the $\bar{x}_{s,t}$ possibilities given a pair of generated sub-paths. However, across iterations, we end up using a naive averaging technique. We could potentially improve the resulting images by performing MIS weighting across iterations, either by weighting all sub-contributions of all iterations simultaneously or via a two-layered MIS weighting scheme.
- **Metropolis Light Transport:** Metropolis Light Transport takes a set of paths, as generated in our BDPT procedure, and performs a set of "mutations" such that the resulting sampling better prioritizes important samples, leading to a more physically accurate simulation.

Bibliography

Dutré, Philip (Sept. 2003). “Global Illumination Compendium”. In: URL: https://www.ii.uni.wroc.pl/~anl/cgfiles/TotalCompendium.pdf.

Potts, Collin and Madelyn Torres (Dec. 2019). “metro-caster”. In: URL: https://github.com/collinsgit/metro-caster.

Veach, Eric (Dec. 1997). “Bidirectional Path Tracing”. In: URL: https://graphics.stanford.edu/courses/cs348b-03/papers/veach-chapter10.pdf.

Vlnas, Michal (Mar. 2018). “Bidirectional Path Tracing”. In: URL: https://cescg.org/wp-content/uploads/2018/04/Vlnas-Bidirectional-Path-Tracing-1.pdf.