Real-time Kd-tree Based Importance Sampling of Environment Maps

Serkan Ergun*
International Computer Institute
Ege University

Murat Kurt†
International Computer Institute
Ege University

Aydın Öztürk‡
Department of Computer Engineering
Yaşar University

Abstract

We present a new real-time importance sampling algorithm for environment maps. Our method is based on representing environment maps using kd-tree structures, and generating samples with a single data lookup. An efficient algorithm has been developed for real-time image-based lighting applications. In this paper, we compared our algorithm with Inversion method [Fishman 1996]. We show that our proposed algorithm provides compactness and speedup as compared to Inversion method. Based on a number of rendered images, we have demonstrated that in a fixed time frame the proposed algorithm produces images with a lower noise than that of the Inversion method. We also demonstrate that our algorithm can successfully represent a wide range of material types.

CR Categories: Computer Graphics [I.3.7]: Three-Dimensional Graphics and Realism—Color, shading, shadowing, and texture

Keywords: environment maps, importance sampling, monte carlo integration, global illumination, rendering, GPU, kd-tree

1 Introduction

Environment maps are commonly used for modeling natural lighting to create realistic images. Complex real-world illumination can be represented efficiently by environment maps. However, high quality rendering of scenes under image-based lighting requires efficient sampling strategies. In this context various sampling strategies have been proposed to reduce the noise in rendered images. The underlying sampling strategies include environment map sampling, bidirectional reflectance distribution function (BRDF) [Nicodemus et al. 1977] sampling, product sampling, and multiple importance sampling (MIS) [Veach 1998].

It has been shown empirically that using neither environment map nor BRDF sampling alone results in images with low noise ratios [Veach 1998]. Product sampling and real-time shading strategies have been proposed to reduce the noise of rendered images. A common drawback of theses strategies is their high computational cost.

On the other hand, the variance of the estimated outgoing radiance can be reduced by using MIS [Veach 1998] which samples environment map and BRDF separately and obtains the probability weighted mixtures of these samples. MIS has been shown to be an efficient strategy for reducing variance in rendered images [Veach 1998]. A natural approach for reducing the computational cost of this rendering procedure is to implement this algorithm on GPU hardware. Often, such procedures require environment map sampling performed on GPU. In this paper, we propose a new environment map sampling algorithm based on kd-trees which can be implemented on modern GPUs.

A commonly used environment map sampling strategy, namely Inversion method [Fishman 1996], has the drawback of using large number of data lookups. On the average, inversion method performs a binary search for each generated sample in $O(\log n)$ data lookups. Our proposed importance sampling method needs a single data lookup and therefore has lower running time for generating samples from environment maps. We empirically show that our algorithm is efficient in real-time rendering.

The rest of this paper is organized as follows. In Section 2, some of the relevant work in sampling from environment maps and real-time shading for image-based lighting are presented. Section 3 describes our approach. Some empirical results are presented in Section 4. The conclusions and future work is given in Section 5.

2 Related Work

Monte Carlo importance sampling has been investigated by computer graphics community for many years [Veach 1998]. Importance sampling is an empirical method of integration to reduce variance by generating samples from a distribution that closely resembles the integrand itself [Pharr and Humphreys 2010]. Environment map sampling, BRDF sampling, product sampling and MIS are well-known importance sampling techniques for rendering scenes under image-based lighting.

Some other methods including stratified sampling [Arvo 2001], hierarchical sampling [Debevec 2005], structured importance sampling [Agarwal et al. 2003], fast blue noise sampling [Ostromoukhov et al. 2004], interleaved sampling [Kollig and Keller 2003], and inversion of the cumulative density function (cdf) [Secord et al. 2002; Lawrence et al. 2005] have also been proposed for environment map sampling. All of these methods are essentially based on generating samples considering the energy distribution of the environment map [Wang and Åkerlund 2009]. Some of these methods [Kollig and Keller 2003; Ostromoukhov et al. 2004; Debevec 2005] generate a sample set in a precomputation step, and use this sample set in rendering. Using a fixed sample set in this way produces banding artifacts for specular materials.

Product of BRDFs and environment maps can be considered to reduce the variance in Monte Carlo integration. This method is known as product sampling. [Burke et al. 2005] proposed a bidirectional importance sampling method for product sampling. Their method is based on *sampling-importance resampling* (SIR) algorithm. Wavelets [Clarberg et al. 2005; Huang et al. 2007] and spherical harmonics [Jarosz et al. 2009] have also been used for product sampling. Although, they provide considerable compression, the computational cost for generating a sample can be high.

*e-mail: serkan.ergun@ege.edu.tr
†e-mail: murat.kurt@ege.edu.tr
‡e-mail: aydin.ozturk@yasar.edu.tr

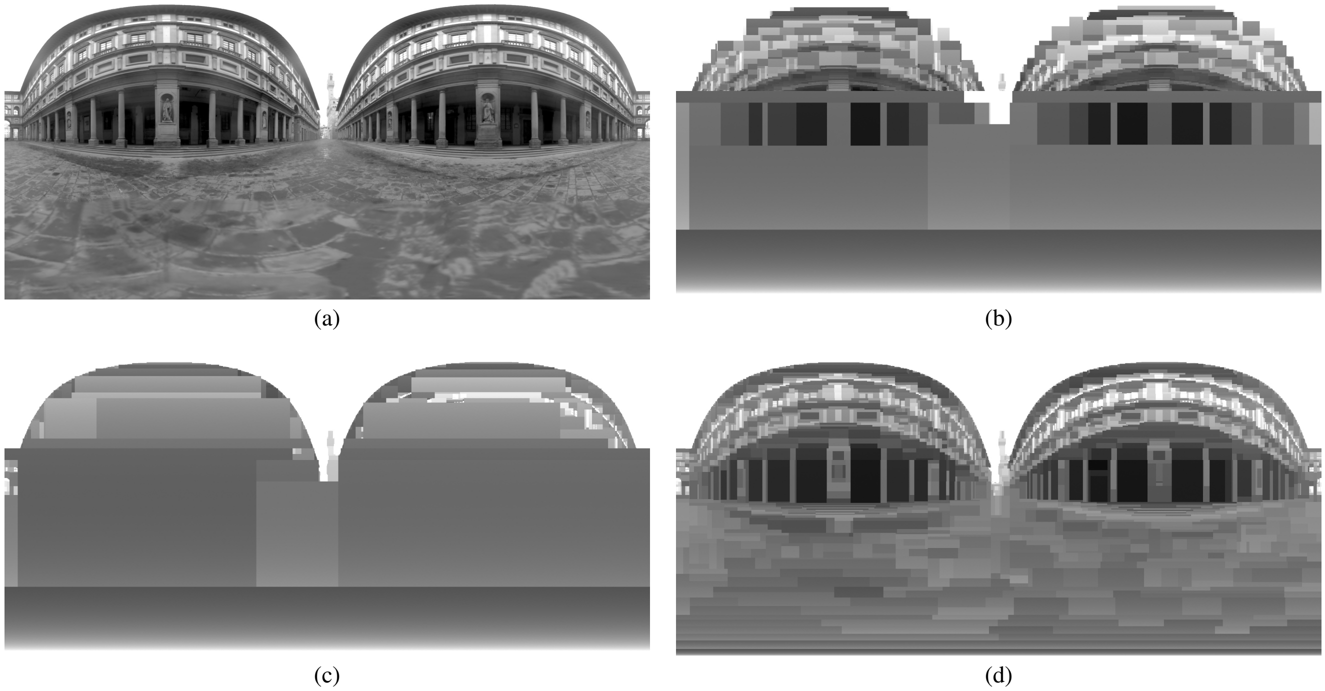


Figure 1: Rendered environment maps (Uffizi) (a): 2048×1024 resolution environment map requiring 8MBs of memory. (b), (c), (d): Same environment map compressed to 48KB (1:170 compression) using range, variance, and SSE criteria, respectively. Environment maps used in this work are a courtesy of Debevec.

Direct illumination from environment maps for real-time applications needs careful attention. Some researchers [Greene 1986; Heidrich and Seidel 1999; Kautz and McCool 2000; Kautz et al. 2000; Ramamoorthi and Hanrahan 2002; Sloan et al. 2002; Ng et al. 2003] have used expensive precomputations, others [Ramamoorthi and Hanrahan 2001; Křivánek and Colbert 2008] have failed to provide an accurate and general method for all material types for rendering scenes interactively. For example, Křivánek and Colbert's method [Křivánek and Colbert 2008] do not represent anisotropic materials properly, Ramamoorthi and Hanrahan's method [Ramamoorthi and Hanrahan 2001] is unable to represent specular materials well.

Kd-tree structure is a well-known data structure used in computer graphics. [McCool and Harwood 1997] used kd-tree structure for importance sampling of conditional distributions such as tabulated BRDF data. They proposed to split the multidimensional function from its center point on each level of the kd-tree. They also proposed a traversal algorithm to generate samples from the conditional probability distribution. On the other hand, our algorithm reduces the error by optimizing the split plane position. While this algorithm runs in $O(log(n))$ time our proposed algorithm requires $O(1)$ time to generate a sample. Kd-tree structures can also be used for ray-tracing, photon-mapping, and k-means clustering [Wang et al. 2009] in computer graphics applications.

The need of using empirical cdfs is unavoidable when dealing with measured illumination [Lawrence et al. 2005]. It is not a trivial process to compress and represent them accurately in an interactive application. A simple solution for real-time environment map sampling could be using inversion method [Fishman 1996] which takes $O(\log n)$ time for each generated sample. Our kd-tree based method generates a sample in $O(1)$ time, providing high compression rates. It is also computationally feasible for real-time rendering and suitable for all material types including isotropic, anisotropic, diffuse, glossy and specular materials.

3 Proposed Method for Environment Map Sampling

An environment map is defined as a $w \times h$ rectangular block. Pixel intensities in this rectangular block can be viewed as sampled probability densities from an unknown bivariate distribution with uniform spacing. In this work, pixel intensities in an environment map are normalized with respect to their block sum so that the volume under the underlying empirical distribution is made equal to 1. Our proposed method is based on splitting this environment map into sub-blocks by using a kd-tree structure. Empirical probabilities corresponding to each sub-block are obtained as sum of the normalized pixel intensities within this sub-block. These empirical probabilities are then sorted in descending order and corresponding block indices are assigned in increasing order. Thus, the plots of these probabilities against sub-block indices can be considered as empirical distribution of these sub-block indices. Representing the empirical distribution of sub-blocks in this way provides a good ground for modeling this empirical distribution by a simple probability density function (pdf). Finally, the resulting estimated model can be used to generate samples for incoming light.

3.1 Kd-tree construction

Kd-tree structure is based on splitting an environment map block recursively. Splitting process is continued until a predetermined number of sub-blocks have been created. Then each pixel intensity

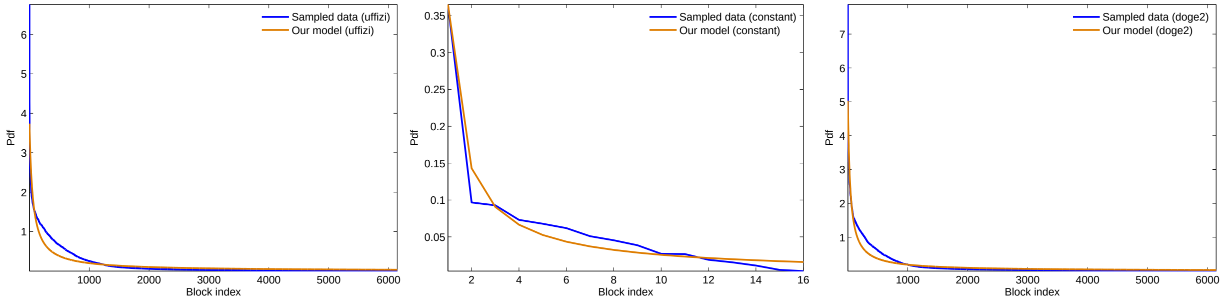


Figure 2: Empirical pdfs of various environment maps and fitted analytical pdfs. Our analytical pdf model can be found in Equation 4.

Algorithm 1 createTree($nBlocks$)

/*f_{ij} *is the 2D environment map block**/
$block \leftarrow f_{ij}$ such that $1 \le i \le w$ and $1 \le j \le h$
$blockList$.add($block$)
for $k = 2 \rightarrow nBlocks$ **do**
  $subBlock \leftarrow$ selectBlock($blockList$)
  $blockList$.remove($subBlock$)
  $[lSubBlock, rSubBlock] \leftarrow$ splitBlock($subBlock$)
  $blockList$.add($lSubBlock$)
  $blockList$.add($rSubBlock$)
end for

Algorithm 2 selectBlock($blockList$)

$maxError \leftarrow 0$
$selectedSubBlock \leftarrow null$
for each $subBlock \in blockList$ **do**
  $error \leftarrow \sum f_{ij}^2 - (\sum f_{ij})^2/(w_b \times h_b)$
  if $maxError < error$ **then**
    $maxError \leftarrow error$
    $selectedSubBlock \leftarrow subBlock$
  end if
end for
return $selectedSubBlock$

within a sub-block is replaced with its sub-block mean. Finally, the original environment map can be reconstructed using these mean pixel intensities in the sub-blocks. The reconstructed environment map is an approximation to the original image and the accuracy of the approximation depends on the predetermined number of sub-blocks to be created.The pseudocode for this procedure is presented in Algorithm 1.

Choosing the most convenient sub-block for splitting during recursion needs a special handling. We proceed to choose the sub-block having the largest intensity variation first. Various measures of variation can be used for this purpose. Results of using three different statistics namely range, variance, and sum of squared error (SSE) for sub-block selection criteria are shown in Figure 1. Based on this special environment map (Uffizi), it is seen that the best result is obtained when SSE is used as selection criterion. This result has also been observed on a number of environment maps.

In this work, we propose to use SSE which is defined by

$$SSE = \sum_{i=1}^{w_b} \sum_{j=1}^{h_b} \left(f_{ij} - \bar{f}\right)^2, \tag{1}$$

as a selection criterion where $\bar{f}$ is the sub block mean, w_b and h_b

Algorithm 3 splitBlock($subBlock$)

$splitF \leftarrow 0$
for $k = 1 \rightarrow w_b$ **do**
  $lSubBlock \leftarrow f_{ij}$ such that $1 \le i \le k$
  $rSubBlock \leftarrow f_{ij}$ such that $k < i \le w_b$
  $lSize \leftarrow lSubBlock.Width \times lSubBlock.Height$
  $rSize \leftarrow rSubBlock.Width \times rSubBlock.Height$
  $lSum \leftarrow$ sum of values in $lSubBlock$
  $rSum \leftarrow$ sum of values in $rSubBlock$
  $F \leftarrow lSum^2/lSize + rSum^2/rSize$
  if $F > splitF$ **then**
    $splitF = F$
    $subBlock1 = lSubBlock$
    $subBlock2 = rSubBlock$
  end if
end for
for $m = 1 \rightarrow h_b$ **do**
  $tSubBlock \leftarrow f_{ij}$ such that $1 \le j \le m$
  $bSubBlock \leftarrow f_{ij}$ such that $m < j \le h_b$
  $tSize \leftarrow tSubBlock.Width \times tSubBlock.Height$
  $bSize \leftarrow bSubBlock.Width \times bSubBlock.Height$
  $tSum \leftarrow$ sum of values in $tSubBlock$
  $bSum \leftarrow$ sum of values in $bSubBlock$
  $F \leftarrow tSum^2/tSize + bSum^2/bSize$
  if $F > splitF$ **then**
    $splitF = F$
    $subBlock1 = tSubBlock$
    $subBlock2 = bSubBlock$
  end if
end for
return $[subBlock1, subBlock2]$

are the sub-block dimensions. This expression can be rewritten as

$$SSE = \sum_{i=1}^{w_b} \sum_{j=1}^{h_b} f_{ij}^2 - \frac{1}{w_b \times h_b} \left(\sum_{i=1}^{w_b} \sum_{j=1}^{h_b} f_{ij} \right)^2. \tag{2}$$

Note that the SSE is the same as the sample variance multiplied by $(w_b \times h_b)$ that is the number of pixels in the sub-block. The main reason of using SSE instead of sample variance as a selection criterion is that it carries information about the underlying sub-block size. The pseudocode of this procedure is presented in Algorithm 2.

Given a sub-block in a kd-tree, the splitting plane position is determined in such a way that the pooled variance [Killeen 2005] of the children blocks is minimum. It can be shown that minimization of the pooled variance can be reduced to maximizing the sum of squares of sub-block totals divided by their respective number of

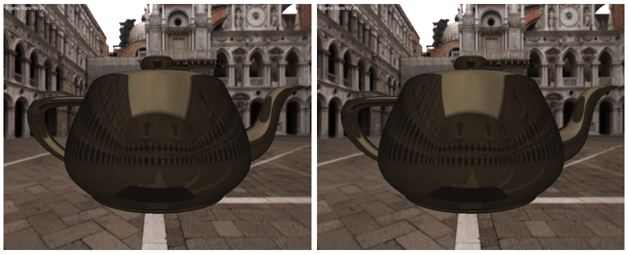


Figure 3: Comparison of inversion method (left) and our method (right) in real-time rendering. In this scene, the chrome-steel teapot has been rendered with both methods using 16 samples/pixel for testing real-time rendering performance.

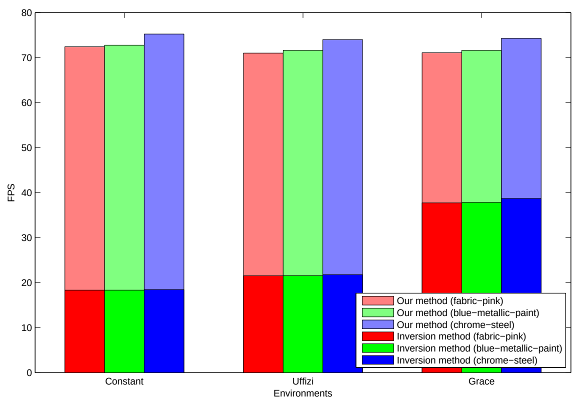


Figure 4: Comparison of our method and inversion method in real-time rendering. Both of the methods were rendered with 16 samples/pixel. The FPS rates are measured under different environment maps.

pixels that is

$$\underset{k}{\operatorname{argmax}} \left(\begin{array}{c} \frac{1}{k}\left(\sum_{j=1}^{h_b}\sum_{i=1}^{k} f_{ij}\right)^2 + \\ \frac{1}{w_b-k}\left(\sum_{j=1}^{h_b}\sum_{i=k+1}^{w_b} f_{ij}\right)^2 \end{array} \right), \tag{3}$$

where w_b and h_b are width and height of the sub-block, respectively. The maximization procedure based on Equation 3 is performed along horizontal edge of the block. A similar procedure should also be carried out along the vertical edge. Pseudocode of this optimization is presented in Algorithm 3.

3.2 Approximation for sub-block distribution

As was mentioned in the previous section, empirical block probabilities are sorted in descending order. Therefore, the corresponding pdf is expected to be an exponential type distribution (see Figure 2). We approximate this pdf by the following monotonically decreasing function

$$p(x) = \frac{1}{\log\left(1+\frac{n}{\alpha}\right)(\alpha+x)}, 0 \le x \le n, \tag{4}$$

where n is the total number of sub-blocks in the kd-tree, and α is the parameter of the distribution. It is easy to show that

$$\int_0^n p(x)dx = 1, \tag{5}$$

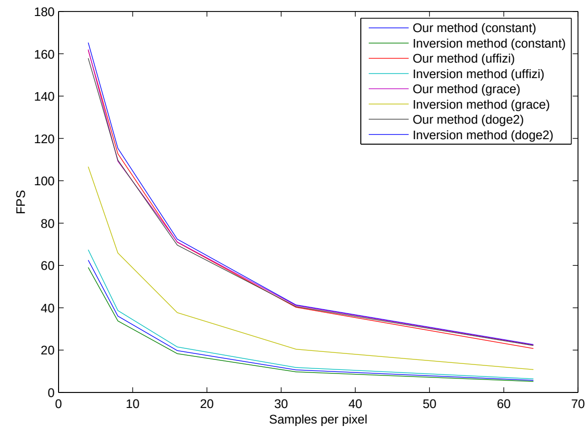


Figure 5: FPS rates of our method and the inversion method for different sample sizes and environment maps.

and the corresponding cdf is

$$P(x) = \frac{\log\left(1+\frac{x}{\alpha}\right)}{\log\left(1+\frac{n}{\alpha}\right)}, \tag{6}$$

where α and n are described in Equation 4.

Various statistical methods can be used to estimate the non-linear parameter α of the distribution. We estimate the parameter using the empirical sub-block distribution. L_1-norm estimate of the parameter is obtained by minimizing the objective function

$$\sum_{x=1}^{n} |\tilde{p}(x) - p(x)|, \tag{7}$$

where x is the index number of the block, n is the number of blocks, $\tilde{p}(x)$ is the empirical pdf, and $p(x)$ is the pdf described in Equation 4. Levenberg-Marquardt algorithm implemented in C/C++ Minpack package [Devernay 2012] is used for minimization of the objective function in Equation 7.

In order to give some insight into the accuracy of the approximation, empirical and fitted pdfs for three different environment maps are obtained and their plots are given in Figure 2. The approximation quality can be improved by introducing more non-linear parameters in the fitted model. However, a pdf with a single parameter has been found to be satisfactory for our sampling procedure.

Pre-processing steps can be summarized as:

1. Kd-tree is constructed using Algorithm 1.
2. In order to form the empirical pdf of the block indices, sub-blocks of the Kd-tree are sorted in descending order based on their average values.
3. The empirical pdf is then approximated by the analytical pdf given in Equation 4. The parameter of this pdf is estimated by using L_1-norm minimization.
4. The pdf parameter and the sub-block bounds are stored to be used later in the sampling procedure.

3.3 Sampling the incoming light

Our importance sampling strategy simply consists of generating sub-block indices first and then generating incoming light direction

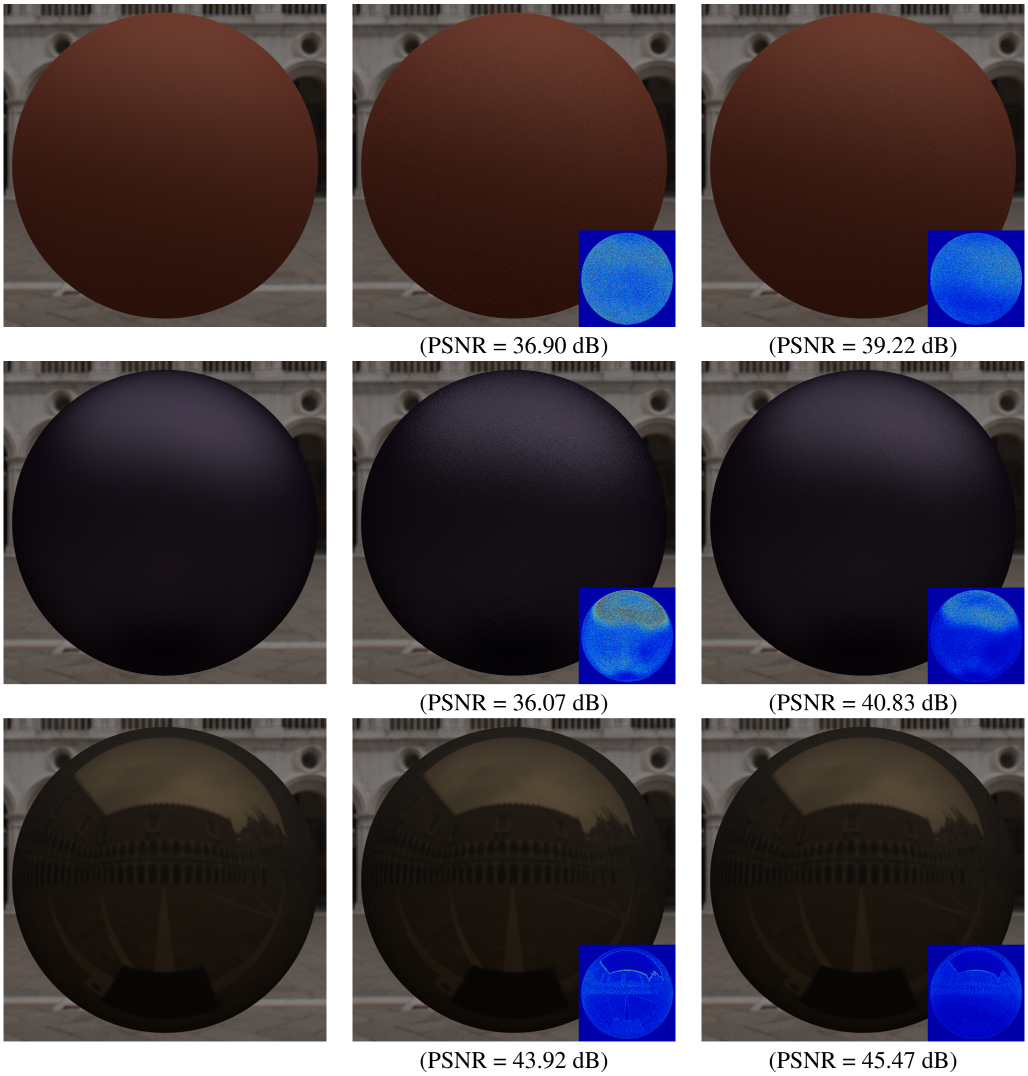


Figure 6: Rendered spheres based on different materials and different sampling methods. Top row: fabric-pink, middle row: blue-metallic-paint, bottom row: chrome-steel materials; first column: reference images were rendered with 512 samples/pixel, second column: images were rendered using inversion method with 8 samples/pixel. third column: images were rendered using kd-tree method with 32 samples/pixel. Insets show the difference between the methods and reference images, and they are scaled by a factor of 5 for higher visibility. PSNR values between the methods and reference images were also shown below each image.

within this block. Once the pdf of the block indices is estimated, then the corresponding cdf is obtained easily. Sub-block indices can be generated using the well-known probability integral transformation method of obtaining random samples from a known distribution. The following inverse function of the cdf based on Equation 6 is used for generating sub-block indices

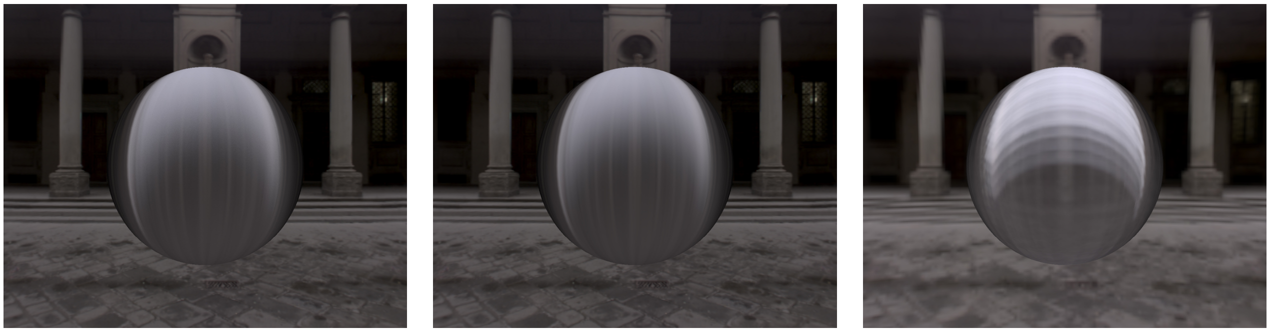
$$x = P^{-1}(\xi) = \alpha\left(\left(1+\frac{n}{\alpha}\right)^{\xi} - 1\right), \tag{8}$$

where α and n are described in Equation 4 and ξ is a uniform (0,1) random variable.

Light direction vectors can be determined randomly by generating the corresponding elevation and azimuth angles. These two variables are not necessarily uniformly and independently distributed in the original environment map. However, they are uniformly distributed within the sub-blocks of the reconstructed environment map since pixel intensities are replaced with their sub-block mean. Therefore, we store the bounds of the sub-blocks in an array and access the corresponding sub-block with the sub-block index generated using Equation 8. Within the bounds of the sub-block, we generate two uniform random variables corresponding to elevation and azimuth angles to obtain a random incoming light direction.

Sampling procedure can be summarized as:

1. Generate three random variables: ξ_1, ξ_2, ξ_3.
2. Select the corresponding block index $x = P^{-1}(\xi_1)$, using Equation 8.
3. Read the bounds of the selected sub-block.
4. Generate elevation and azimuth angles uniformly within the bounds of the selected sub-block using ξ_2, ξ_3.
5. The probability of this sample can be computed with $p(x)/Area(\text{sub-block})$.



(a) (b) (c)

Figure 7: Rendered spheres using anisotropic Ward BRDF model with parameters $\alpha_x = 0.5, \alpha_y = 0.001$. (a) Reference image, (b) our kd-tree based importance sampling method (c) Křivánek and Colbert's real-time filtered importance sampling method.

4 Results

We employed the MIS strategy to investigate some empirical properties of our kd-tree based importance sampling procedure. The MIS strategy requires generating samples both from a BRDF model and from an environment map. In this work, Ward model [Ward 1992] is employed to represent BRDF. Parameters of Ward model have been obtained by Ngan et al. [Ngan et al. 2005] for a number of diffuse and specular materials. We have used their published results for this model. Samples from an environment map were generated following the procedures explained in Section 3. To compare our method with its competitors, we have implemented a commonly used sampling method, namely inversion method [Fishman 1996]. Křivánek and Colbert's implementation of their method [Křivánek and Colbert 2008] has also been included in the comparison.

Real-time rendering implementations of our method and inversion method [Fishman 1996] were made using OpenSceneGraph [Burns and Osfield 2004], NVIDIA CUDA [NVIDIA 2012], and Random123 [Salmon et al. 2011] libraries. Our method, and the inversion method [Fishman 1996] were implemented using Physically Based Rendering Toolkit (PBRT) [Pharr and Humphreys 2010] for off-line renderings. All programs were executed on an Intel Core i7-920 (2.67 GHz) with 12GBs of RAM and NVIDIA GeForce GTX 480 GPU.

Renderings of a teapot object based on inversion method and our method are presented in Figure 3. The images in the figure are the screenshots of a dynamic scene rendered using inversion and our method with 19 and 72 frames per second (FPS), respectively.

Three environment maps and three material types are used to compare the methods in terms of frame rates. The empirical results are illustrated in Figure 4. Scenes including a teapot object are rendered using 16 samples per pixel. It is seen in the figure that our method has higher FPS values than that of inversion method [Fishman 1996] in all cases. The low performance of inversion method is due to the fact that it performs data lookups in $O(\log(n))$ while our method performs a single data lookup for a single sample. As is seen in Figure 4, inversion method performs poorly especially on environment maps having small variation. It is interesting to note that even the performance of our method slightly depends on the BRDF types, it does not depend on the type of environment map used.

To investigate the relative performance of the methods under different sample sizes, we rendered a teapot object with $n = 8, 16, 32$ and 64 samples per pixel for three different environment maps. The results are shown in Figure 5. It is seen that our method has higher performance for all sample sizes. The effect of different environment maps on the performance of the inversion method [Fishman 1996] is clearly observed.

We made a visual comparison of the methods by fixing the rendering times at 40 FPS and rendered spheres under direct illumination using Doge2 [Debevec 1998] environment map. The rendering results are shown in Figure 6. The Peak Signal-to-Noise Ratio (PSNR) [Richardson 2002] values for each rendered sphere were calculated and shown on the figure. In this special implementation, our method has uniformly yielded higher PSNR values. This expected situation is due to the higher number of samples generated by our method than that of inversion method [Fishman 1996] within a fixed time frame.

Since our kd-tree based importance sampling method is a direct implementation of Monte Carlo importance sampling, it can represent every material type accurately. In order to demonstrate this situation, we rendered spheres in real-time using anisotropic Ward BRDF model [Ward 1992] with parameters $\alpha_x = 0.5, \alpha_y = 0.001$. We compared our method with that of GPU-based real-time shading method proposed by Křivánek and Colbert [Křivánek and Colbert 2008]. As shown in Figure 7, our method represents anisotropic materials accurately. Křivánek and Colbert reported that their method may not represent anisotropic materials efficiently, and they left removing this limitation as a future work.

An attractive property of our method is that it has the flexibility of controlling the number of sub-blocks to be used. To increase the compression ratio of an environment map, we need to decrease the number of the underlying sub-blocks. Environment maps having small variations generally require small number of sub-blocks at a given quality level. Therefore, higher compression ratios can be achieved for this kind of environment maps. In our case, to fit the environment map data into GPU constant buffer, we selected 6144, 6144, 16 as the number of sub-blocks for Uffizi, Doge2, and Constant environment maps, respectively. The compression ratios of our method relative to inversion method were found to be $1:170, 1:170$, and $1:65536$ for Uffizi, Doge2, and Constant environment maps, respectively.

5 Conclusions and Future Work

In this work, a kd-tree based method for importance sampling of environment maps has been proposed. We compared our method with a well-known importance sampling method, namely the Inversion method. We compared both methods on different platforms, such as CPU and GPU. Real-time rendering times (FPS values), image qualities (PSNR values), and compression ratios were obtained for comparison by using different environment maps. We have empirically showed that our method has outperformed the inversion method in terms of real-time rendering times, image qualities, and compression ratios.

We also compared our method with a real-time GPU-based importance sampling method proposed by Křivánek and Colbert. Based on a single material, we demonstrated that our method can also be used for anisotropic materials, and it can represent the material accurately under direct illumination.

As a future work, kd-tree sampling of other multi-dimensional functions such as BRDF will further be investigated.

Acknowledgments

Authors would like to thank anonymous reviewers for their constructive comments and Paul Debevec for the HDR environment maps used in this work.

References

AGARWAL, S., RAMAMOORTHI, R., BELONGIE, S., AND JENSEN, H. W. 2003. Structured importance sampling of environment maps. *ACM Trans. Graph. 22* (July), 605–612.

ARVO, J. 2001. Stratified sampling of 2-manifolds. In *State of the Art in Monte Carlo Ray Tracing for Realistic Image Synthesis, SIGGRAPH 2001 Course Notes*, vol. 29.

BURKE, D., GHOSH, A., AND HEIDRICH, W. 2005. Bidirectional importance sampling for direct illumination. In *Proceedings of the Eurographics Symposium on Rendering*, Eurographics Association, 147–156.

BURNS, D., AND OSFIELD, R. 2004. Open scene graph a: Introduction, b: Examples and applications. In *Proceedings of the IEEE Virtual Reality 2004*, IEEE Computer Society, Washington, DC, USA, 265.

CLARBERG, P., JAROSZ, W., AKENINE-MÖLLER, T., AND JENSEN, H. W. 2005. Wavelet importance sampling: efficiently evaluating products of complex functions. *ACM Trans. Graph. 24* (July), 1166–1175.

DEBEVEC, P. 1998. Rendering synthetic objects into real scenes: bridging traditional and image-based graphics with global illumination and high dynamic range photography. In *Proceedings of the 25th annual conference on Computer graphics and interactive techniques*, ACM, New York, NY, USA, SIGGRAPH '98, 189–198.

DEBEVEC, P. 2005. A median cut algorithm for light probe sampling. In *ACM SIGGRAPH 2005 Posters*, ACM, New York, NY, USA, SIGGRAPH '05.

DEVERNAY, F., 2012. C/C++ Minpack, January. http://devernay.free.fr/hacks/cminpack/cminpack.html.

FISHMAN, G. S. 1996. *Monte Carlo: Concepts, algorithms, and applications*. Springer Series in Operations Research. Springer-Verlag, New York.

GREENE, N. 1986. Environment mapping and other applications of world projections. *IEEE Comput. Graph. Appl. 6* (November), 21–29.

HEIDRICH, W., AND SEIDEL, H.-P. 1999. Realistic, hardware-accelerated shading and lighting. In *Proceedings of the 26th annual conference on Computer graphics and interactive techniques*, ACM Press/Addison-Wesley Publishing Co., New York, NY, USA, SIGGRAPH '99, 171–178.

HUANG, H.-D., CHEN, Y., TONG, X., AND WANG, W.-C. 2007. Incremental wavelet importance sampling for direct illumination. In *Proceedings of the 2007 ACM symposium on Virtual reality software and technology*, ACM, New York, NY, USA, VRST '07, 149–152.

JAROSZ, W., CARR, N. A., AND JENSEN, H. W. 2009. Importance sampling spherical harmonics. *Computer Graphics Forum 28*, 2, 577–586.

KAUTZ, J., AND MCCOOL, M. D. 2000. Approximation of glossy reflection with prefiltered environment maps. In *In Graphics Interface*, 119–126.

KAUTZ, J., VÁZQUEZ, P.-P., HEIDRICH, W., AND SEIDEL, H.-P. 2000. Unified approach to prefiltered environment maps. In *Proceedings of the Eurographics Workshop on Rendering Techniques 2000*, Springer-Verlag, London, UK, 185–196.

KILLEEN, P. R. 2005. An alternative to null-hypothesis significance tests. *Psychol Sci 16*, 5, 345–353.

KOLLIG, T., AND KELLER, A. 2003. Efficient illumination by high dynamic range images. In *Proceedings of the 14th Eurographics workshop on Rendering*, Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, EGRW '03, 45–50.

KŘIVÁNEK, J., AND COLBERT, M. 2008. Real-time shading with filtered importance sampling. *Computer Graphics Forum 27*, 4, 1147–1154.

LAWRENCE, J., RUSINKIEWICZ, S., AND RAMAMOORTHI, R. 2005. Adaptive numerical cumulative distribution functions for efficient importance sampling. In *Proceedings of the Eurographics Symposium on Rendering*, Eurographics Association, O. Deussen, A. Keller, K. Bala, P. Dutré, D. W. Fellner, and S. N. Spencer, Eds., 11–20.

MCCOOL, M. D., AND HARWOOD, P. K. 1997. Probability trees. In *Graphics Interface'97*, 37–46.

NG, R., RAMAMOORTHI, R., AND HANRAHAN, P. 2003. All-frequency shadows using non-linear wavelet lighting approximation. *ACM Trans. Graph. 22* (July), 376–381.

NGAN, A., DURAND, F., AND MATUSIK, W. 2005. Experimental analysis of brdf models. In *Proceedings of the Eurographics Symposium on Rendering*, Eurographics Association, 117–226.

NICODEMUS, F. E., RICHMOND, J. C., HSIA, J. J., GINSBERG, I. W., AND LIMPERIS, T. 1977. Geometrical considerations and nomenclature for reflectance. *Science And Technology 160*, October, 1–52.

NVIDIA, 2012. CUDA introduction page, January. http://developer.nvidia.com/category/zone/cuda-zone.

OSTROMOUKHOV, V., DONOHUE, C., AND JODOIN, P.-M. 2004. Fast hierarchical importance sampling with blue noise properties. *ACM Trans. Graph. 23* (August), 488–495.

PHARR, M., AND HUMPHREYS, G. 2010. *Physically Based Rendering, Second Edition: From Theory To Implementation*, 2nd ed. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

RAMAMOORTHI, R., AND HANRAHAN, P. 2001. An efficient representation for irradiance environment maps. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, ACM, New York, NY, USA, SIGGRAPH '01, 497–500.

RAMAMOORTHI, R., AND HANRAHAN, P. 2002. Frequency space environment map rendering. *ACM Trans. Graph. 21* (July), 517–526.

RICHARDSON, I. E. 2002. *Video Codec Design: Developing Image and Video Compression Systems*. John Wiley & Sons, Inc., New York, NY, USA.

SALMON, J. K., MORAES, M. A., DROR, R. O., AND SHAW, D. E. 2011. Parallel random numbers: as easy as 1, 2, 3. In *Proceedings of 2011 International Conference for High Performance Computing, Networking, Storage and Analysis*, ACM, New York, NY, USA, SC '11, 16:1–16:12.

SECORD, A., HEIDRICH, W., AND STREIT, L. 2002. Fast primitive distribution for illustration. In *Proceedings of the 13th Eurographics workshop on Rendering*, Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, EGRW '02, 215–226.

SLOAN, P.-P., KAUTZ, J., AND SNYDER, J. 2002. Precomputed radiance transfer for real-time rendering in dynamic, low-frequency lighting environments. *ACM Trans. Graph. 21* (July), 527–536.

VEACH, E. 1998. *Robust monte carlo methods for light transport simulation*. PhD thesis, Stanford, CA, USA. AAI9837162.

WANG, R., AND ÅKERLUND, O. 2009. Bidirectional importance sampling for unstructured direct illumination. *Computer Graphics Forum 28*, 2, 269–278.

WANG, R., WANG, R., ZHOU, K., PAN, M., AND BAO, H. 2009. An efficient gpu-based approach for interactive global illumination. *ACM Trans. Graph. 28*, 3 (July), 91:1–91:8.

WARD, G. J. 1992. Measuring and modeling anisotropic reflection. In *Proceedings of the 19th annual conference on Computer graphics and interactive techniques*, ACM, New York, NY, USA, SIGGRAPH '92, 265–272.