Stratified Sampling in Metropolis Light Transport

Tom Jankauski

May 12, 2015

Abstract

Stratified sampling is a common method in Monte Carlo integration techniques to reduce variance. Stratified sampling has been difficult to implement in Metropolis Light Transport, as each sample is a mutation of the previous sample. This mutation can cause the algorithm to sample outside of the stratum, and cause the algorithm to no longer be stratified. This paper presents a theoretical framework for adapting Metropolis Light Transport to be stratified. It then discusses a practical example of stratifying over the surface of a light in a scene. Finally, it discusses practical concerns of stratifying with Metropolis Light Transport.

1 Introduction

Monte Carlo based rendering algorithms attempt to produce an image by integrating a rendering equation. Monte Carlo methods make use of random samples to perform the integration. The random sampling can lead to variance in the samples and shows up as noise in the final image.

Reducing variance is one way to produce a more clear image with the same amount of computation. Stratification is a common technique to reduce the variance of a random sample. Stratification divides the space of all samples into many different disjoin sample spaces, called strata. The total number of samples taken is then divided among the strata and each of the stratum are sampled independently. This ensures a more equal representation of each stratum, and helps to reduce the variance of the sampling.

2 Theory

The Metropolis Light Transport (MLT) algorithm integrates the contribution that a light path makes to the final image over the space of all possible

light paths in a scene. To generate the next sample, the current path is mutated. With a certain probability, this mutated sample is accepted and added to the image. The acceptance probability is calculated so the probability of sampling a value, x, is proportional to the importance of x, called $I(x)$.

$\bar{x} \leftarrow$ InitialPath()
for $i \leftarrow 1 to N$ **do**
  $y \leftarrow$ mutate(x)
  $a \leftarrow$ acceptanceProb($y|x$)
  if *random()* $< a$ **then**
    $x \leftarrow y$
  end
  add sample x to the image
end

def *acceptanceProb*($y|x$)**:**
  return $\min(\frac{I(y)T(y \to x)}{I(x)T(x \to y)}, 1)$
end

Algorithm 1: A basic outline of MLT[1]

Where $I(x)$ is the importance function, and $T(x \to y)$ is the probability of mutating from x to y.

New samples in MLT come from mutation of previous samples. The rules that govern how samples are mutated must be carefully designed to explore important paths. In practice, it can be difficult to develop a mutation function that only produces samples within a given stratum.

Hoberock's paper[2] on arbitrary importance functions showed that the importance function can be designed to enforce a sampling density. This can be used to ensure that we pull samples only from the current stratum.

3 General Stratification

Let Ω be the space of all possible inputs to the function. Let $f : \Omega \to \mathbb{R}$ be the function we are integrating. $I : \Omega \to \mathbb{R}^+$ is an existing importance function.

Ω is divided in to the strata $\Omega_1 \dots \Omega_k$, which are mutually disjoint. We can then create an importance function for each stratum as follows,

$$I_j(x) = \begin{cases} I(x) & \text{if } x \in \Omega_j \\ 0 & \text{otherwise} \end{cases}$$

Such an importance function ensures that no samples are taken from outside Ω_j. Any sample that lies outside of Ω_j has an importance of 0, so the acceptance probability of that sample will also be 0. This enforces stratification, and is typically easy to design. The only requirement is a function that can tell if a given sample lies within a given stratum.

With this, we can run the stratified sampling algorithm as follows.

function *StratifiedMetropolis*(f, *numSamples*)**:**
    sample $\leftarrow$ empty sample
    for $i \leftarrow 1$ *to* k **do**
        $w_i \leftarrow$ weight of stratum Ω_i
        $s_i \leftarrow w_i \times$ numSamples
        Metropolis$_i$(sample, f, s_i)
    end
    return sample
end

Algorithm 2: Stratified Metropolis

Where Metropolis$_i$(sample, f, s_i) deposits s_i samples to the cumulative sample (named sample), according to importance function I_i

4 Stratifcation Over Light Sources

Stratifying over the area of light sources is a common stratification for bidirectional path tracing. It reduces variance for pixels that are influenced differently by different areas of the light. This section details how we can use stratified MLT to stratify over the lights, and addresses some of the practical concerns of doing so.

4.1 Selecting Strata

For metropolis light transport Ω is the space of all possible light paths in the scene. $\bar{x} \in \Omega$ is a path that starts at a light source, makes some number of bounces in the scene, and ends at the camera. $f(\bar{x})$ is the contribution that $\bar{x}$ makes to the image. This is the power of light that starts at the light source and travels along $\bar{x}$ to the camera. $I(\bar{x})$ can be any reasonable importance function. A typical function would be the luminance of $f(\bar{x})$.

To create the strata, subdivide the surface of each light in the scene. Each subdivision of each light will become its own stratum. In this case, we divide each rectangular light into several smaller rectangles, but any polygonal division will work. Let Ω_i be all light paths that start at subdivision i of

the light. The importance function is then defined as

$$I_i(\bar{x}) = \begin{cases} I(\bar{x}) & \text{if } \bar{x} \text{ begins at light } i \\ 0 & \text{otherwise} \end{cases}$$

The endpoint of $\bar{x}$ is determined when the path is first generated. Generally, this is found when a ray is cast and hits a light source or when a sample is generated starting at a light source. This information can be tagged onto $\bar{x}$ to avoid repeating calculation.

This algorithm then stratifies samples over the surface of the light sources in the scene.

4.2 Sampling Efficiency

Although the samples are already stratified, the mutation function can also be changed to increase sampling efficiency and reduce variance. Without these modifications, time would be wasted generating samples that are completely rejected. At the same time, because the importance function ensures stratification, we do not need to change all of the mutation functions.

For this implementation, the only part of the mutation function changed was bidirectional mutation. Bidirectional mutation mutates path $\bar{x}$ by deleting a subpath from $\bar{x}$ and tracing a new subpath from the scene. When this mutation deletes the start of the path, we simply change it to select a new starting point on the current light. Now bidirectional mutations will always generate paths in the current stratum.

The other components of the mutation function were left as is. Some of the mutation functions would always mutate to a path in the current stratum if the original path was in the current stratum. This was the case with lens perturbation, which never move the start of the path. Other mutations tended to make small changes to the path, and simply didn't move the path off of the current light stratum that often.

With these small changes to the mutation function, the algorithm can stratify over light sources with very few wasted samples.

5 Results

To measure performance, a high quality of the scene was taken as a baseline image. Then shorter renders, taking one tenth the number of samples, were rendered for both stratified and unstratified algorithms. The first measure

is the average of the relative difference of the luminance of the pixels.

$$\text{relativeDiff}(a, b) = \frac{(a-b)^2}{a}$$

Where a is the accepted value, and b is the value of the test render. For relative difference, a smaller value is better. If the images are identical, then the relative difference will be 0.

The second measure percent of mutations that were accepted. Having a low acceptance rate means the samples are heavily correlated. This produces an image with high variance and appears in the image as noise. A higher acceptance rate is better.

Each of these scores were averaged over 10 renders of each algorithm.

Scene	Algorithm	Relative Difference	Acceptance Rate
Ring	Stratified	0.010	34.61%
	Unstratified	0.013	34.40%
Cornell Box	Stratified	0.026	42.35%
	Unstratified	0.025	42.10%
Susuan	Stratified	0.070	33.88%
	Unstratified	0.080	33.82%

The stratified sampling method showed improvement over standard metropolis sampling. It showed better performance in producing a closer image to the final image, and had the same or better acceptance rate. The image quality can be best seen in the accuracy of the shadows and in diffuse surfaces.

Because of the changes to bidirectional mutations, there was no significant loss in acceptance rate. The experiments imply that the stratified method actually has an improved acceptance rate. This may be because bidirectional mutations make a smaller change to the current path. This would mean the mutated path has a better chance of being of a similar importance to the original path. This leads to the path having a higher chance of being accepted.

Stratification over light sources showed the greatest improvements in scenes with large light sources and multiple light sources. Stratification allowed each part of the light to have an equal representation in the final render and reduced noise in the image.

Here we have some examples of the test renders. In figure 1, the stratified render reduces the noise over the diffuse surfaces in the scene. It also improves the quality of the shadow.

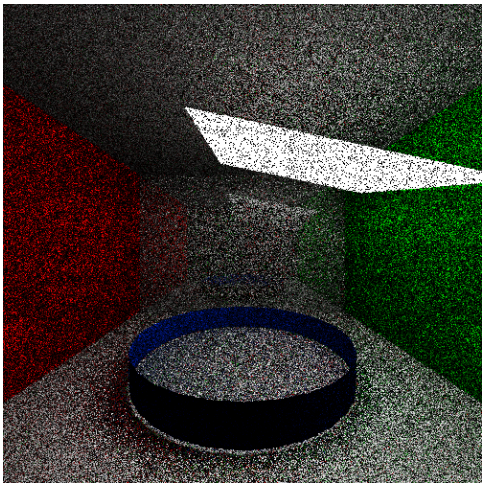

Figure 1: Stratified

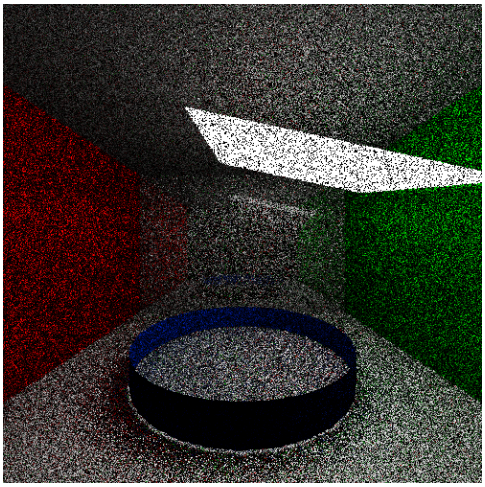

Figure 2: Unstratified

6 Possible Extensions

6.1 Improving Sample Acceptance

Stratification over lights can lead to samples being completely rejected when they otherwise would not have. Experimentally, this is unlikely and will not change the performance in most scenes. One possible modification is to only use the updated bidirectional mutation rule without the change to the

importance function. This no longer stratifies the samples, but can still get a better representation of the light source. This may get similar results to stratification without the risk of rejecting additional samples.

6.2 Updating Weights of Strata

Initially the weight of each stratum is set by an estimation of how much it contributes to the scene. The power that a section of light produces is a good estimate. We can, however, iteratively update the weight of each stratum based on its actual contribution to the image. This can be done by dividing the sampling into rounds. On the first round, take a stratified random sample with an initial estimate for the weights of the strata. With those samples, measure the actual contribution of the strata to the image, and change the weights to reflect the contribution.

References

[1] Veach, E. and Guibas, L., *Metropolis Light Transport* SIGGRAPH (1997)

[2] Hoberock, J. and Hart, J., *Arbitrary Importance Functions for Metropolis Light Transport* Computer Graphics Forum (2010)