7-1-2007

# Light-Based Sample Reduction Methods for Interactive Relighting of Scenes with Minute Geometric Scale

William B. Kerr
*Dartmouth College*

Fabio Pellacini
*Dartmouth College*

# LIGHT-BASED SAMPLE REDUCTION METHODS FOR INTERACTIVE RELIGHTING OF SCENES WITH MINUTE GEOMETRIC SCALE

William B. Kerr Fabio Pellacini

Dartmouth College
Dartmouth Computer Science Technical Report TR2007-600

**ABSTRACT**

*Rendering production-quality cinematic scenes requires high computational and temporal costs. From an artist's perspective, one must wait for several hours for feedback on even minute changes of light positions and parameters. Previous work approximates scenes so that adjustments on lights may be carried out with interactive feedback, so long as geometry and materials remain constant. We build on these methods by proposing means by which objects with high geometric complexity at the subpixel level, such as hair and foliage, can be approximated for real-time cinematic relighting. Our methods make no assumptions about the geometry or shaders in a scene, and as such are fully generalized. We show that clustering techniques can greatly reduce multisampling, while still maintaining image fidelity at an error significantly lower than sparsely sampling without clustering, provided that no shadows are computed. Scenes that produce noise-like shadow patterns when sparse shadow samples are taken suffer from additional error introduced by those shadows. We present a viable solution to scalable scene approximation for lower sampling reolutions, provided a robust solution to shadow approximation for sub-pixel geomery can be provided in the future.*

## 1 Introduction

Cinematic relighting presents the problem of receiving fast and reliable feedback from a renderer when lights in a three dimensional scene have changed. For production quality scenes, changing a parameter on even one light can require the scene to be fully re-rendered, forcing the lighting artist to wait several hours for feedback in some cases. Other options include substituting simplified versions of the geometry and shaders into the scene to reduce render time, but the quality reduction of this approach makes it applicable only to preliminary blocking phases of light design and not to more detailed refinement. Rapid feeback is critical to an artist's workflow, and several relighting techniques have been proposed to handle this by approximating the scene while maintaining accuracy with respect to its reproduction.

We propose methods for reducing the number of samples taken from scenes that include geometric elements smaller than the size of a single pixel. Examples of such elements include thin hairs and foliage with small leaves, stems, or blades of grass. Scenes like these pose problems for current relighting approaches, since sparsely sampled deep framebuffers and blurring cause detail at the subpixel level to be lost. Our approach investigates the merging of samples at a subpixel level into a reduced set of samples that maintain the original level of detail.

To do this, we organize the material and intersection data provided by a sample from the scene into a generalized feature vector. Our methods operate on these vectors without any semantic knowledge or assumptions about the data they carry. Once the reduction has been performed, we are able to restore meaning to the vector through a categorical typing system set up by the artist. Essentially, all parameters within a sample are quantized into a vector with a type, clustered using one of several methods, and then reinterpreted once the vectors have been merged and effectively reduced.

Our work trades static geometry and materials for flexibility with respect to lighting. We assume a fixed viewing angle with no changes to any element in the scene other than lights. Since the lighting equation can be computed quickly and efficiently, we preprocess the scene and store information about geometry and materials. Any light parameters, including position, may be changed without referring back to the original set of samples. If geometry or material parameters are changed, however, we must preprocess again before relighting. For this paper, we both preprocess and render using software. We claim that once the samples have been reduced to a manageable size, translating to a hardware implementation that runs at interactive rates can be achieved using methods similar to those found in [Pellacini et al. 2005].

## 2 Methodology

In order to relight a scene in substantially less time than the full render, we work on the assumption that there exists a reduced set of data that can be processed by the renderer, producing very near the same result as the full set of data. We choose to reduce the amount of samples actually rendered through the lighting equation after intersections

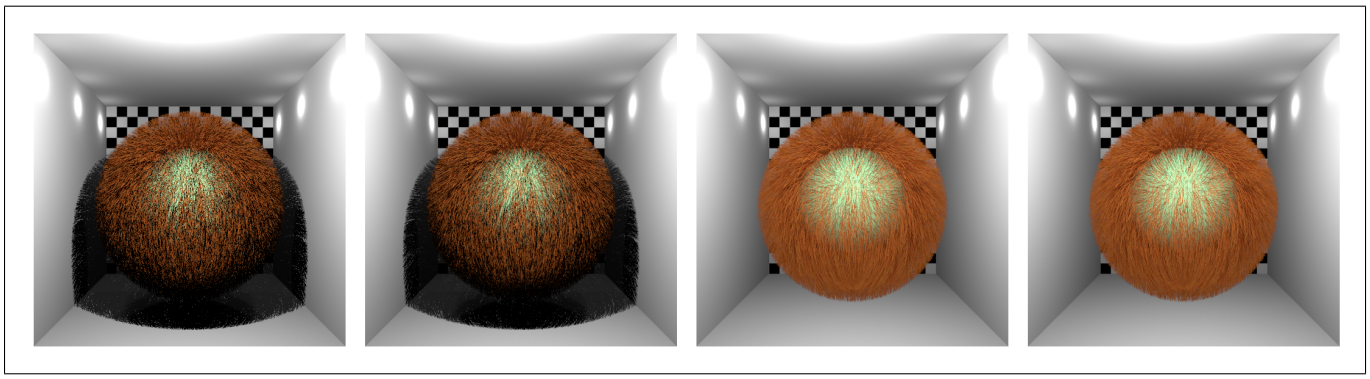

Figure 1: From left to right: A) Image sampled at 25spp with shadows, B) reduced to 8spp with K-means method, C) sampled at 25spp without shadows, and D) reduced to 8spp with K-means method without shadows.

or rasterizations are calculated. Clustering techniques allow us to reduce the number of samples by merging similar data points from the full set of samples. From this point forward, the notion of a sample will be taken from our ray-tracing implementation. Note that these methods are generally applicable to any kind of scene samples, as opposed to just ray intersections.

## 2.1 Samples

Our approach relies on quantizing samples generated by the ray tracer into multidimensional vectors that can be clustered. For every intersection of a view ray with the scene, all necessary data required to render the color value at that intersection is stored in a sample and made available in the form of a vector $\alpha$. The data collected need not be the same for every sample, but only sample vectors of the same dimensionality with parallel representation can be clustered together. For example, one sample may contain a position, normal, and diffuse coefficient, whereas another may contain an additional specular coefficient, making it impossible to merge the two. Furthermore, if the first sample has an additional luminosity coefficient, even though the two sample vectors share the same dimensionality, they would not have parallel representation. We leave it to the shader artist to construct materials in type groups that can be clustered homogeneously. The resulting clustered sample is then represented as another vector, $\bar{\alpha}$, with the same dimensionality. This vector provides the renderer with all information necessary to compute the lighting equation.

When merging samples, there exists data for which it makes sense to average all merged values and data for which a single representative must be chosen. For example, a shader parameter such as a diffuse coefficient would be averaged across all samples being merged. However, one sample out of those being merged would be chosen as a representative for intersection position to guarantee a position on the surface of an object for lighting. Our methods do not rely on the semantic meaning of data elements within a sample, and as such, require an artist to define which elements are to be averaged and which are to be representative.

## 2.2 K-means Approach

Once the vectors for all samples are built, we carry out a clustering procedure to determine which samples to merge together. The choice of k-means clustering as a simple, intuitive method presents itself as a standard to benchmark against for more advanced methods. All samples are calculated and clustered using the k-means algorithm [Mac67] without any lighting knowledge. The samples in each cluster are then merged. We choose the sample closest to the cluster centroid as the representative for all data elements not averaged. Our goal for the project is to outperform the bar set by this trivial approach.

## 2.3 Light Metric

We experiment on multiple variations of a light-based metric designed to perform clustering in a way more suited to the specific problem of relighting. A light metric operates on the notion that some samples have more influence than others over the color of an entire pixel when light configurations vary. Our goal is to find samples that minimize the error function, $\epsilon$, over the multiple integral representing all possible parameter configurations of a general light type.

$$\epsilon = \int_{\lambda \in L} \left( \frac{\sum_{i=1}^{s} f(\{\alpha_i\}, \{\lambda\})}{s} - f(\{\bar{\alpha}\}, \{\lambda\}) \right)^2 \quad (1)$$

where $L$ is all possible light configurations, $s$ is the number of samples in the pixel, and $f$ outputs the rendered color from the given light and sample

We approximate the integral in equation 1 using Monte Carlo over a set of $l$ lights with randomized parameters.

$$\epsilon = \frac{1}{l}\sum_{j=1}^{l}(\frac{\sum_{i=1}^{s} f(\{\alpha_i\},\{\lambda_j\})}{s} - f(\{\bar{\alpha}\},\{\lambda_j\}))^2 \quad (2)$$

We expect the lighting artist to provide bounds on the lighting parameters for his or her given scene. For example, the valid range of positions a point light can assume.

## 3 Implementation

We implement k-means clustering as well as six versions of our light metric clustering. Each implementation consists of three steps: choosing representatives, clustering, and merging. Every pixel of $s$ samples is reduced to $k$ samples by one of the following methods. Since samples of differing material types cannot be merged from our general perspective, each method precalculates, based on the ratio of each material type in the pixel, a subset $k_t$ of the $k$ resulting samples for each material type. The number of reduced samples of each type is directly proportional to the number of original samples of each type. We give each merged sample a weight based on the number of original samples belonging to its cluster, where

$$weight = \frac{samples\ in\ cluster}{samples\ in\ all\ clusters} \quad (3)$$

Note that some samples may be lost due to underrepresented material types that are not given a cluster. Unless otherwise noted, all calculations are performed within the scope of a single pixel.

### 3.1 K-means

Our K-means implementation chooses representatives after the clustering has been performed. The first $k$ samples are chosen as initial centroids, after which the clusters are recalculated until convergence is reached. Once the clustering is complete, the sample closest to the cluster centroid in parameter space by Euclidean distance is chosen as the representative. All parameters not indicated as representative are averaged across all samples in the cluster. This technique uses no information about possible lighting conditions, and merges based on intersection and material parameters alone.

### 3.2 Light Metric 1

All four Light Metric implementations begin by choosing representative samples based on minimizing equation 2. We parameterize lights in the same way as materials so that any number of random lights can be generated by randomly interpolating between two representative lights, which are passed in by an artist. Our implementation keeps an $n$x$l$ data structure of the rendered color value of each sample on each random light. These values are calculated for each light individually, as if it was the only light in the scene.

To avoid exponential computational costs, instead of computing equation 2 for every possible clustering, we approximate it through representative samples. For each material type, Light Metric 1 chooses the $k_t$ samples that, when rendered alone, most closely match the rendered output of the entire pixel. I.e., for every sample $\alpha$ of material type $t$, we calculate values for $\epsilon_\alpha$ such that

$$\epsilon_\alpha = \frac{1}{l}\sum_{j=1}^{l}(\frac{\sum_{i=1}^{s} f(\{\alpha_i\},\{\lambda_j\})}{s} - f(\{\alpha\},\{\lambda_j\}))^2 \quad (4)$$

The $k_t$ samples with minimum $\epsilon_\alpha$ are chosen as representatives and become initial cluster centroids. We determine cluster membership for the rest of the samples of the given material type by assigning samples to the cluster whose representative centroid is closest to it by Euclidean distance. Centroids are not recalculated. Once clusters are built, Light Metric 1 merges each of them into a single sample as in K-means.

### 3.3 Light Metric 2

Light Metric 2 works in much the same way as Light Metric 1, with the addition of shader groups. When choosing representatives, Light Metric 1 does not take into account that even though a group of samples may have the minimum $\epsilon_\alpha$ values when evaluated individually, together they may skew the pixel output towards a majority material parameter configuration. For example, if the pixel contains 6 blue phong samples and 2 red phong samples, and 4 representatives are chosen, then all 4 representatives will probably be blue. A more accurate group of representatives would probably be 3 blue and 1 red.

After the value of each sample is evaluated for each random light, Light Metric 2 does the following. First, the samples are partitioned into shader groups and each shader group is assigned a probability. Each material type has its own independent set of shader groups. We determine shader group membership using only the average color value for a sample across all random lights. The algorithm begins by making the first sample its own shader group, represnted by a color value. It iterates through the remaining samples and either adds a sample to an existing shader group by averaging the color value of the sample into the shader group, or creates a new shader group with the sample's color value. To determine whether sample $A$ will be a member of shader group $B$, where the RGB color values of both are treated as vectors, we compute

*$cos\theta = A \cdot B/(|A|\,|B|)$*
*Euclidean distance $A - B$*
*If ($\theta < 30^o$) and ($A - B < \sqrt{3}/5$)*
*Then A is considered a member of B and is averaged in*

*Else A is a new shader group*

Once all samples belong to a shader group, we calculate $P_{SG}$ for every shader group. $P_{SG}$ is the probability that the shader group will be chosen to provide the next representative.

$$P_{SG} = \frac{samples\ in\ shader\ group}{samples\ in\ all\ shader\ groups} \tag{5}$$

Like Light Metric 1, Light Metric 2 chooses representatives for cluster centroids using equation 4. For every material type, instead of choosing the minimum $k_t$ samples from all samples of this type, we follow the procedure

1. Select a shader group based on $P_{SG}$
2. Add the sample in this shader group that has minimum $\epsilon_\alpha$ to the list of representatives
3. Remove this sample from the shader group
4. Repeat from 1 until all $k_t$ representatives are chosen

The subsequent clustering and merging operations are carried out as in section 3.2.

## 3.4 Light Metric 3

This implementation puts a greater emphasis on diffuse lighting when choosing representatives. It carries out exactly as in section 3.3 through shader group calculations and clustering. Once the clustering is complete, and before merging, we recalculate the rendered value of each random light on each sample, treating it as a fully white Lambert material with no shadows. Using these new values, we choose the sample from each cluster with the minimum $\epsilon_\alpha$, and make it the representative for that cluster. Merging is then carried out as usual.

## 3.5 Light Metric 4

Light Metric 4 addresses the same problem that shader groups address in section 3.3. Essentially, rather than choosing each sample that minimizes $\epsilon_\alpha$ individually, we choose the group of samples that minimize collectively. Light Metric 4 accomplishes this with a greedy approach.

We proceed as in section 3.2, until it is time to choose representatives. The sample that minimizes $\epsilon_\alpha$ becomes the first representative. Let $V$ be the $n$x$l$ array that stores the rendered value of each sample on each random light. We store an array, $R$, of length $l$ that parallels a row of $V$. $R$ represents the conglomeration of the representative samples collected thus far. We initialize $R$ to the values in the row of $V$ for the first representative. When searching for the remaining representatives we use a modified version of equation 4:

$$\epsilon_{\alpha_R} = \frac{1}{l}\sum_{j=1}^{l}(\frac{\sum_{i=1}^{s} f(\{\alpha_i\},\{\lambda_j\})}{s} - f(\{\alpha_R\},\{\lambda_j\}))^2 \tag{6}$$

where $\alpha_R$ is the value of $R$ if $\alpha$ was averaged in. I.e., $f(\{\alpha_R\},\{\lambda_j\}) = R_j$

For each of the $k_t$ representatives, the sample, $\alpha$, that minimizes equation 6 and has not already been averaged into $R$ becomes the next representative. It is then averaged into $R$, updating the values in $R$.

Once all representatives are chosen, clustering and merging proceed as in section 3.2.

## 3.6 Light Metric 5

Light Metric 5 produces the same $n$x$l$ array $V$ as menioned in section 3.5. It then treats each of the $n$ rows of $V$ as the feature vector for the sample it represents. For example, if rgb color is being used, then the feature vector would have $3l$ components. These feature vectors are then used by the K-means algorithm to cluster, choose representatives, and merge as in section 3.1.

## 3.7 Light Metric 6

Light Metric 6 clusters exactly the same way as K-means in section 3.1. It then chooses representatives as in Light Metric 1 using the method from section 3.2 and equation 4. Merging occurs as in section 3.2. Light Metric 6 illustrates the influence representatives have over the discrepancy between K-means and Light Metric 1.

# 4 Results

We conduct several experiments to assess the performance of each clustering method by comparing the images it produces to the fully sampled image, images purely sampled at reduced rates, and images with samples reduced by the other clustering methods. Experiment 1 presents a comparison of performance for all methods at a fixed reduction rate of 25 to 8spp. Experiments 2 and 3 explore the effects of varying reduction rates and the number of random lights used by the Light Metrics respectively. For any of the reduced images $I_r$, we define its error with respect to the original, fully sampled image $I_o$ as follows:

$$Error = \frac{1}{n}\sum_{i=1}^{n}\sqrt{\frac{(Rd_i)^2 + (Gd_i)^2 + (Bd_i)^2}{3}} \tag{7}$$

where $Rd_i$, $Gd_i$, and $Bd_i$ are the red, green, and blue values from the $i$th pixel in the difference image $I_d = I_r - I_o$, and $n$ is the number of pixels in $I_d$.

The "'no shadows"' experiment trials perform sample reduction in exactly the same way as their shadowed counterparts, but do not compute shadows when the resulting image is rendered. Sample reduced images without shadows are compared to fully sampled images without shadows.

## 4.1 Scenes

To test the effectiveness of our approach, we use two scenes with geometry smaller or thinner than the size of a single pixel. The first consists of a ball covered in hair that follows the Kajiya model [Kajiya and Kay 1989?], figure 8. The second consists of geometry taken from the PBRT plant ecosystem scene. This scene, shown in figure 9, features subpixel detail in masses of foliage.

## 4.2 Experiment 1

We carry out the following experiment to test the effectiveness of each implementation from section 3 as the scenes are relit. For each of 50 trials:

1. Light the scene with a single, randomly placed, omnidirectional point light.
2. Render the scene without sample reduction (PURE) at 25spp.
3. Render the scene without sample reduction (PURE) at 8spp.
4. Reduce the number of samples per pixel from 25 to 8 using each of the 7 clustering implementations and render.
5. Record the error: average difference in a color channel between the reduced image and the full image.

When reducing samples for each scene, we use 200 random lights for our Light Metric clustering. We configure the lights so that position is the only parameter randomized. A summary of the experiment can be seen in figure 2.

Throughout the experiment, K-means produces lower error than any of the four Light Metric implementations. Light Metric 4 produces the lowest error on the hair scene, and Light Metric 1 produces the lowest error on the plant scene out of all Light Metric implementations.

In the case of the hair scene with shadows, pure sampling at 8spp outperforms all clustering methods. However, when shadows are turned off when rendering the sample-reduced images, pure sampling falls behind every clustering method. Pure sampling results in almost twice as much error as even the lowest performing Light Metric in the case of no shadows. The shadows computed for the rendered images at 8spp do not take into account any approximation of the scene's geometry. As such, they are simply determined from a simple visibility calculation at 8spp. On the hair scene, this produces shadows that closely resemble noise because there is a low probability that any of the 8 shadow samples will detect a small hair obscuring light for a pixel. The noise introduced by these shadows masks

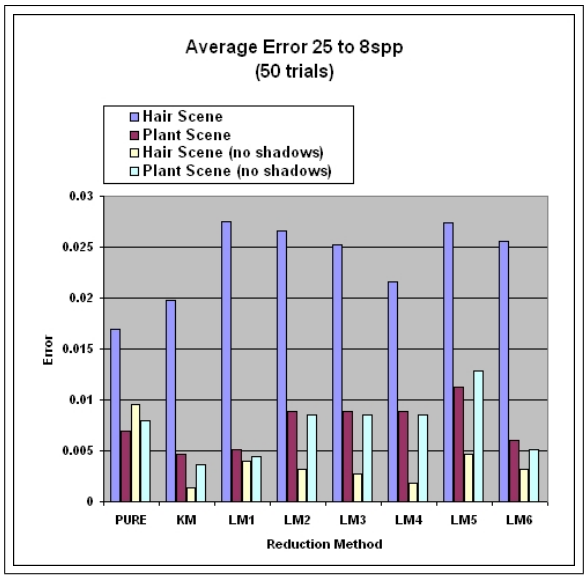


Figure 2: Results from experiment 1.

over the smoothing and approximation introduced by our clustering methods. To help illustrate this concept, imagine two art forgers have a contest to see who can reproduce Leonardo da Vinci's Mona Lisa more exactly. Both are skilled painters, but one captures the subtleties of da Vinci's style to perfection whereas the other can only produce a general visage of the original. To an expert, the first painter would be an obvious winner, but before the paintings are to be judged someone sprinkles sand over both paintings, obscuring their details. The shadows in our images have the same effect as the sand. Even though one image is truer to the original in most respects, noise obscures the details that make it so.

Results from a single trial of experiment 1 can be seen in figure 10. The sampling rate of 25spp is used as a base since this is the minimum number of samples needed to render a single hair continuously across several pixels. Notice that even at 25spp, shadows across the hairs appear slightly irregular. At 8spp, much of the shadowing within the hair cloud has been reduced to noise (see image B from figure 10.

While K-means, Light Metric 1, and Light Metric 6 produce less error than pure sampling in the plant scene for both shadowed and unshadowed trials, Light Metrics 2-5 do not. Through several experiments we have determined that Light Metric 5 does not perform well in general. On the other hand, the results for Light Metrics 2-4 seem to contradict results from the hair scene. For example, why does Light Metric 4 perform so poorly on the plant scene, but excel above the other Light Metrics on the hair scene? Light Metric 1 chooses the closest representatives to the average across all random lights, so it does a better job of . Conversely, Light Metrics 2-4 introduce error across smooth curved surfaces because they work to preserve heterogeneity in the cluster representatives. Figure 3

illustrates the shortcoming of Light Metric 4 compared to pure sampling on a scene with only very large objects with smooth surfaces. Both K-means and Light Metric 1 reduce the sphere scene in figure 3 with 0.0 error.

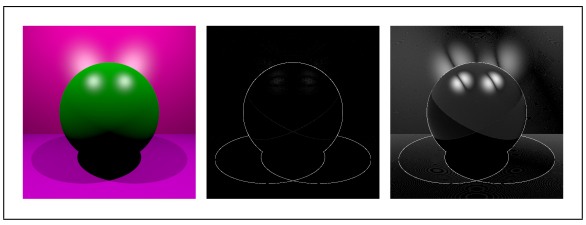

Figure 3: From right to left: A) Base sphere scene rendered at 25spp, B) Error image by pixel for pure sampling at 2spp (10x intensity), C) Error image by pixel for reduction to 2spp by Light Metric 4 (10x intensity)

Results from a single trial of experiment 1 can be seen in figure 12. Notice that shadows black out a large portion of the image with very few artifacts even at 8spp. This accounts for the increase in error for the rendered scenes with shadows turned off. The shadows obscure much of the areas where discrepancies could occur.

## 4.3 Experiment 2

Given the result from Experiment 1 in the case that pure sample reduction outperforms clustering approaches, we conduct a second experiment to observe what happens when we reduce the sampling rate even further. For this experiment, we reduce our test set to pure sampling and the best performing clustering methods from experiment 1: K-means, Light Metric 1, and Light Metric 4. We render both the hair scene and the plant scene with the lighting configurations seen in figures 8 and 9 at 25spp. We then reduce the number of samples to 8, 4, and 2. Results on images rendered with shadows and without shadows (ns) can be seen in figures 4 and 5

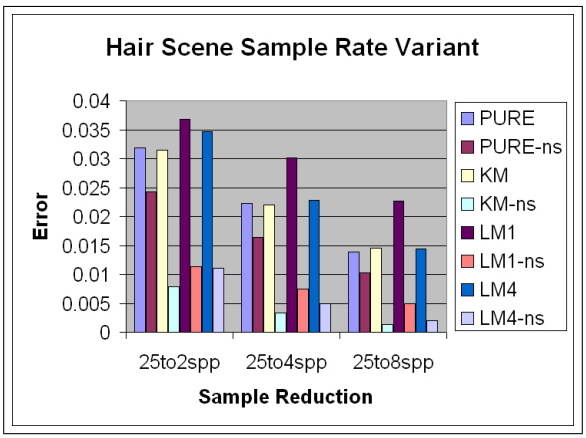


Figure 4: Results of experiment 2 on the hair scene. (ns = no shadows)

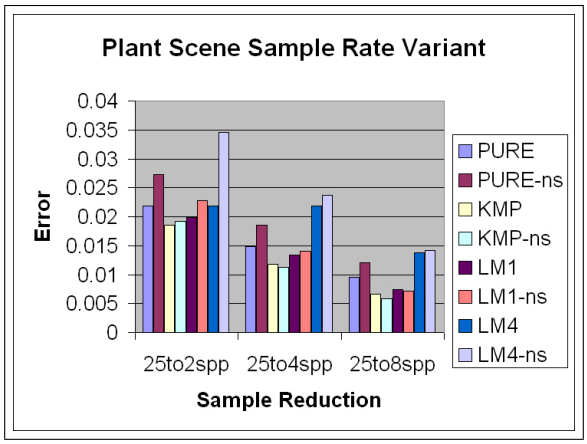


Figure 5: Results of experiment 2 on the plant scene. (ns = no shadows)

## 4.4 Experiment 3

We run an experiment to evaluate the fidelity of our Light Metric methods as the number of random lights used to cluster is increased. K-means outperforms the Light Metric implementations given the number of random lights used in the trials in experiment 1, but could the Light Metric do better? We evaluate ony Light Metrics 1 and 4 for this comparison since they produce the lowest error for the hair and plant scenes respectively. Experiment 2 compares the error from K-means on the hair scene to Light Metrics 1 and 4 using 12, 25, 50, 100, 200, 400, and 800 random lights for cluster precomputation. The results can be seen in figures 6 and 7.

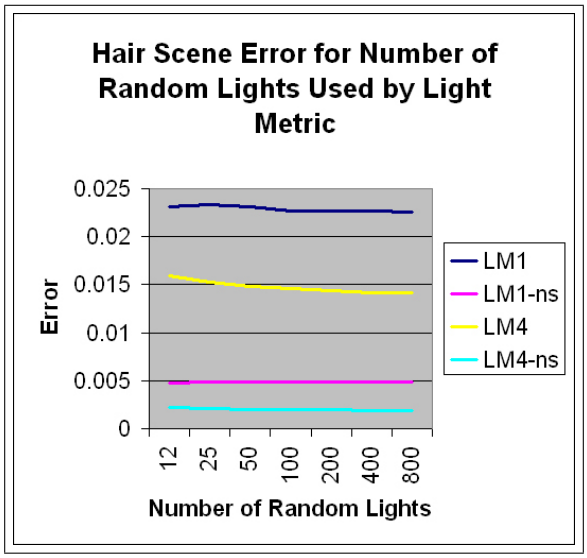


Figure 6: Results of experiment 3 on the hair scene. (ns = no shadows)

Increasing the number of random lights beyond 12 has little effect on the performance of the Light Metrics. The

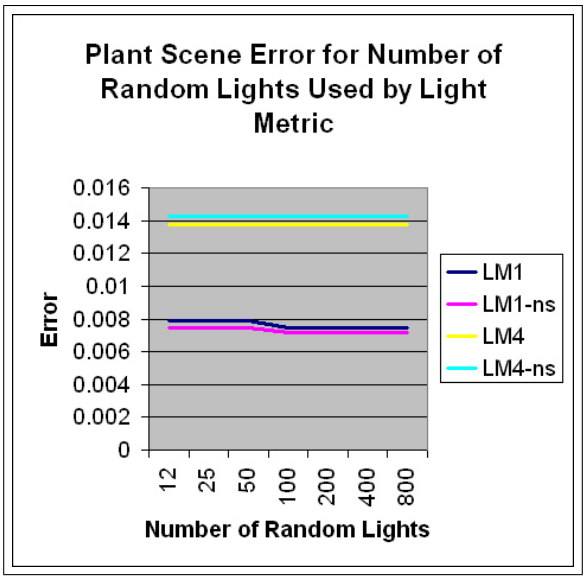


Figure 7: Results of experiment 3 on the plant scene. (ns = no shadows)

performance difference that does result from varying light count is negligible and only spans about a 0.003 range. The slight increase in error for the hair scene with no shadows is most likely attributed to noise introduced by each random light from shadows (figure 6). We do not see this effect as heavily with shadows because the shadow noise in the final rendered image masks over it.

Experiment 3 concludes that K-means yields lower error than our Light Metric implementations regardless of the number of random lights that are used. This implies that the fundamental approach to approximating equation 2 used for the Light Metric implementations produces images with lower fidelity to a full render than the blind clustering generated by K-means.

## 5 Conclusions and Future Work

In the case of scenes with intricate geometry at the sub-pixel level, we have shown that a clustering approach to sample reduction can produce approximations with greater fidelity to the original image than purely rendering the scene with a lower number of samples. There are cases where pure sample reduction actually produces lower error than clustering, but this can be attributed to noise in the shadow signal at the lower sample rate. When shadows produce enough noise from the sub-pixel geometry to mask over the smoothing that clustering provides, the benefits of clustering are nullified. This can be seen in the descrepancy between experiments run with and without shadows. In the case where pure sample reduction outperforms clustering with shadows, it falls short of clustering when shadows are not present. The error discrepancy between the two is much greater in the unshadowed case.

Our methods cannot approximate highly sampled shadows due to the arbitrary position from which visibility must be calculated when lights are moved. Unless a light-obscuring point lies on the surface of an object that can be seen in the projected view plane, the preprocessed clusters have no knowledge of it. This suggests the need for an independent solution to approximating the shadow signal. The challenge here involves sampling the shadow signal at a reduced rate and maintaining the same shadowed to unshadowed sample ratio as if an infinite number of samples were taken. This way, very thin geometry would cast continuous shadows, eliminating the noise artifacts suffered in these experiments.

Our results indicate that a light-based approach may be inferior to a simple data clustering algorithm when reducing the number of samples for relighting. This outcome seems unlikely given the nature of the minimization function given in equation 2. We believe that our rough approximation of the Monte Carlo equation is to blame, but this begs the question of how to achieve a more accurate approximation.

We observe no significant performance increase as the number of random lights used in the Light Metric implementations increases. No matter how many lights are used, K-means will still produce lower error, calling into question why such a light-ignorant clustering method would produce such impressive results.

The strength of the generalized clustering approach, implemented here with K-Means, comes from the lack of assumptions made about the scene. We originally hypothesized that this aspect would be a weakness compared to the Light Metric, since the evaluation of the scene with random lighting provides our Light Metric with more information by which to cluster. The additional information provided by the random lights could still prove useful in some other form, but our research shows that using it to cluster around centroids chosen from samples that produce rendered values closest to the value of the entire pixel does not increase performance.

Future avenues of research on this topic would include investigating other generalized clustering algorithms such as C-Means as well as better ways to cluster samples so that equation 1 is approximated more closely and efficiently.

## References

[HPB06] Miloš Hašan, Fabio Pellacini, and Kavita Bala. Direct-to-indirect transfer for cinematic relighting. In *SIGGRAPH '06: ACM SIGGRAPH 2006 Papers*, pages 1089–1097, New York, NY, USA, 2006. ACM Press.

[KGPB05] Jaroslav Křivánek, Pascal Gautron, Sumanta Pattanaik, and Kadi Bouatouch. Radiance caching for efficient global illumination computation. *IEEE Transactions on Visualization and Computer Graphics*, 11(5), September/October 2005. Also available as Technical Report #1623, IRISA, http://graphics.cs.ucf.edu/RCache/index.php.

[KK89] J. T. Kajiya and T. L. Kay. Rendering fur with three dimensional textures. In *SIGGRAPH '89: Proceedings of the 16th annual conference on Computer graphics and interactive techniques*, pages 271–280, New York, NY, USA, 1989. ACM Press.

[Mac67] J. B. Macqueen. Some methods of classification and analysis of multivariate observations. In *Proceedings of the Fifth Berkeley Symposium on Mathemtical Statistics and Probability*, pages 281–297, 1967.

[MJC+03] Stephen R. Marschner, Henrik Wann Jensen, Mike Cammarano, Steve Worley, and Pat Hanrahan. Light scattering from human hair fibers. In *SIGGRAPH '03: ACM SIGGRAPH 2003 Papers*, pages 780–791, New York, NY, USA, 2003. ACM Press.

[MM06] Jonathan T. Moon and Stephen R. Marschner. Simulating multiple scattering in hair using a photon mapping approach. In *SIGGRAPH '06: ACM SIGGRAPH 2006 Papers*, pages 1067–1074, New York, NY, USA, 2006. ACM Press.

[NRH03] Ren Ng, Ravi Ramamoorthi, and Pat Hanrahan. All-frequency shadows using non-linear wavelet lighting approximation. In *SIGGRAPH '03: ACM SIGGRAPH 2003 Papers*, pages 376–381, New York, NY, USA, 2003. ACM Press.

[PVL+05] Fabio Pellacini, Kiril Vidimče, Aaron Lefohn, Alex Mohr, Mark Leone, and John Warren. Lpics: a hybrid hardware-accelerated relighting engine for computer cinematography. In *SIGGRAPH '05: ACM SIGGRAPH 2005 Papers*, pages 464–470, New York, NY, USA, 2005. ACM Press.

[WABG06] Bruce Walter, Adam Arbree, Kavita Bala, and Donald P. Greenberg. Multidimensional lightcuts. In *SIGGRAPH '06: ACM SIGGRAPH 2006 Papers*, pages 1081–1088, New York, NY, USA, 2006. ACM Press.

[WFA+05] Bruce Walter, Sebastian Fernandez, Adam Arbree, Kavita Bala, Michael Donikian, and Donald P. Greenberg. Lightcuts: a scalable approach to illumination. In *SIGGRAPH '05: ACM SIGGRAPH 2005 Papers*, pages 1098–1107, New York, NY, USA, 2005. ACM Press.

[WTL06] Rui Wang, John Tran, and David Luebke. All-frequency relighting of glossy objects. *ACM Trans. Graph.*, 25(2):293–318, 2006.

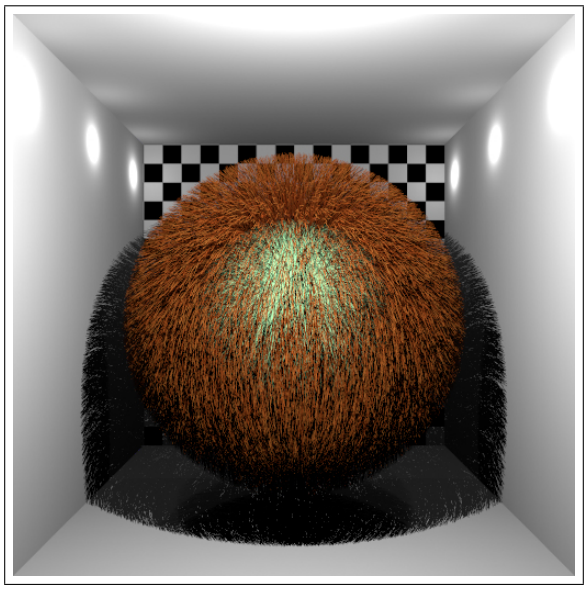

Figure 8: Hair scene, 25 samples per pixel

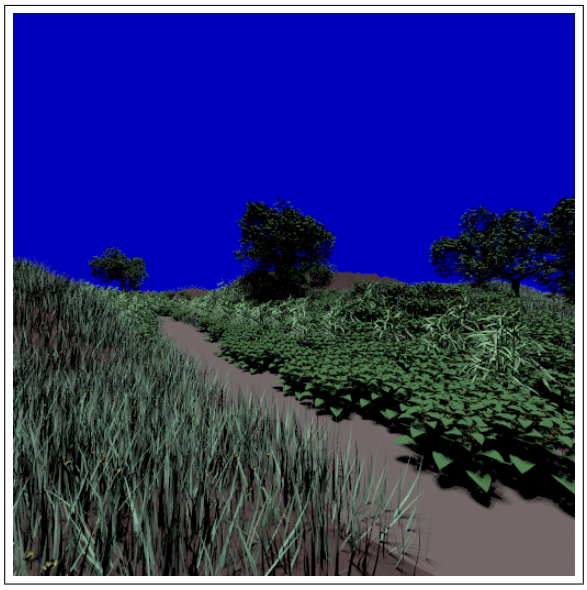

Figure 9: Plant scene, 16 samples per pixel

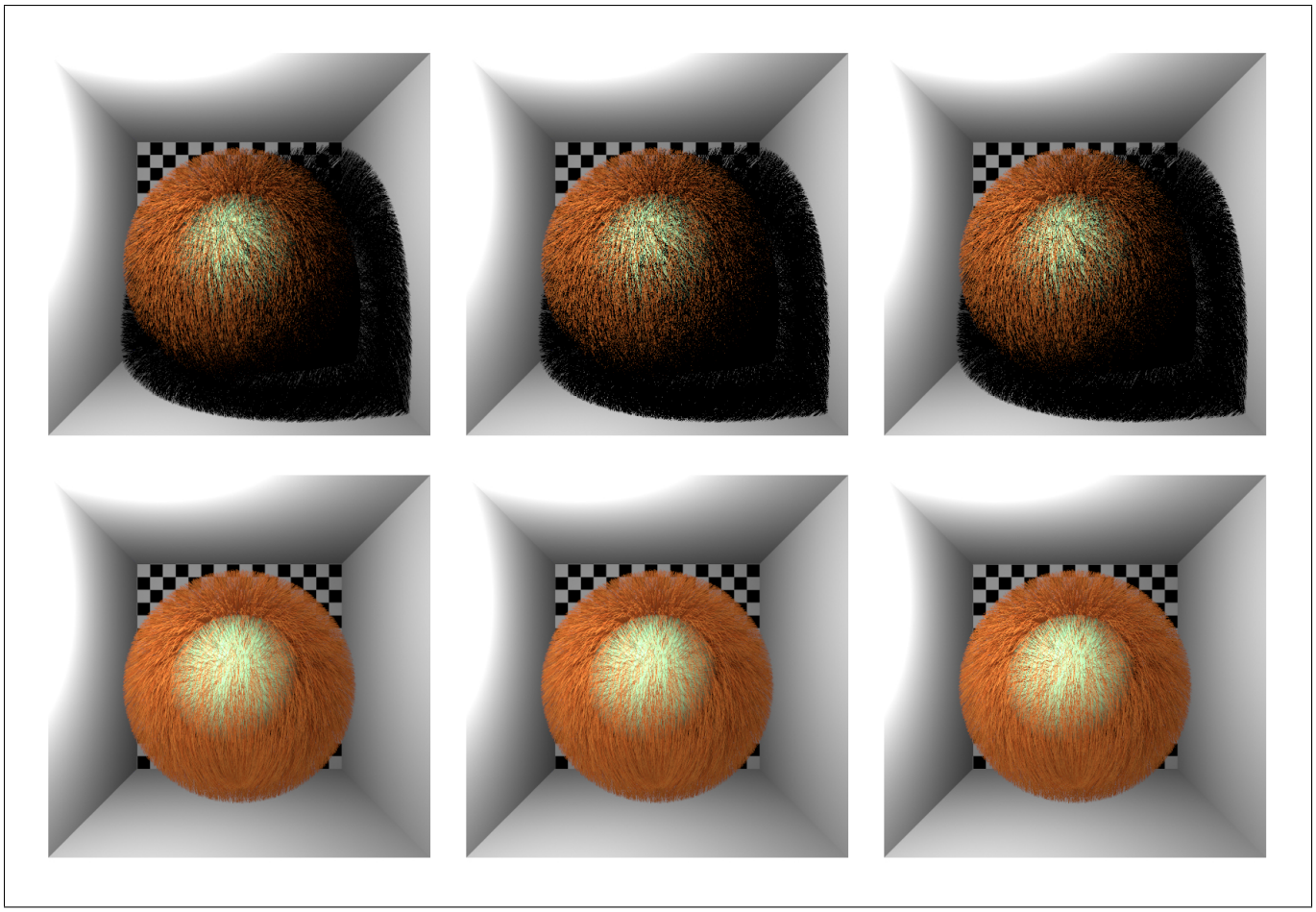

Figure 10: Single trial from experiment 1 on hair scene. From left to right: A) original 25spp image, B) K-means reduced to 8spp, C) purely sampled 8spp. Top row: with shadows. Bottom row: without shadows.

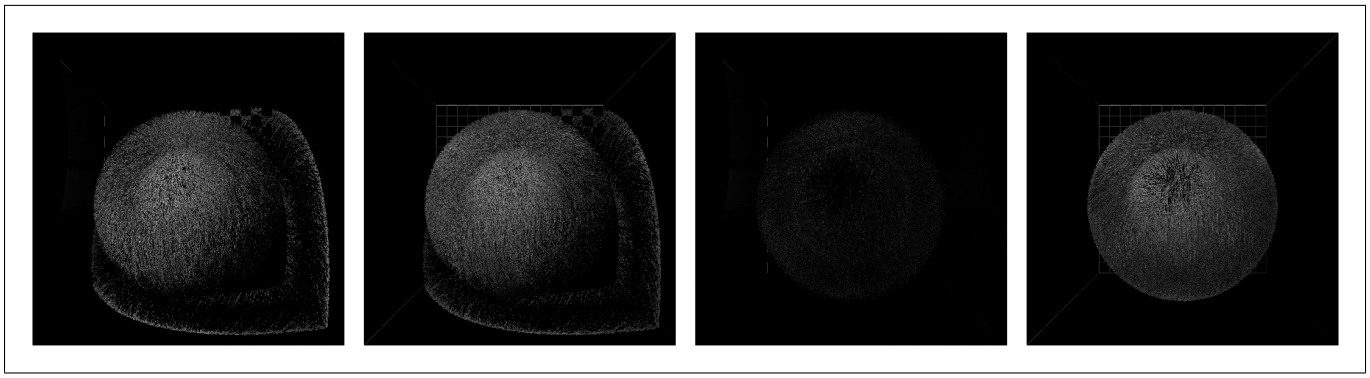

Figure 11: Images displaying error perpixel for single trial from experiment 1 on hair scene. From left to right: A) K-means reduced to 8spp with shadows, B) purely sampled 8spp with shadows, C) K-means reduced to 8spp without shadows, D) purely sampled 8spp without shadows.

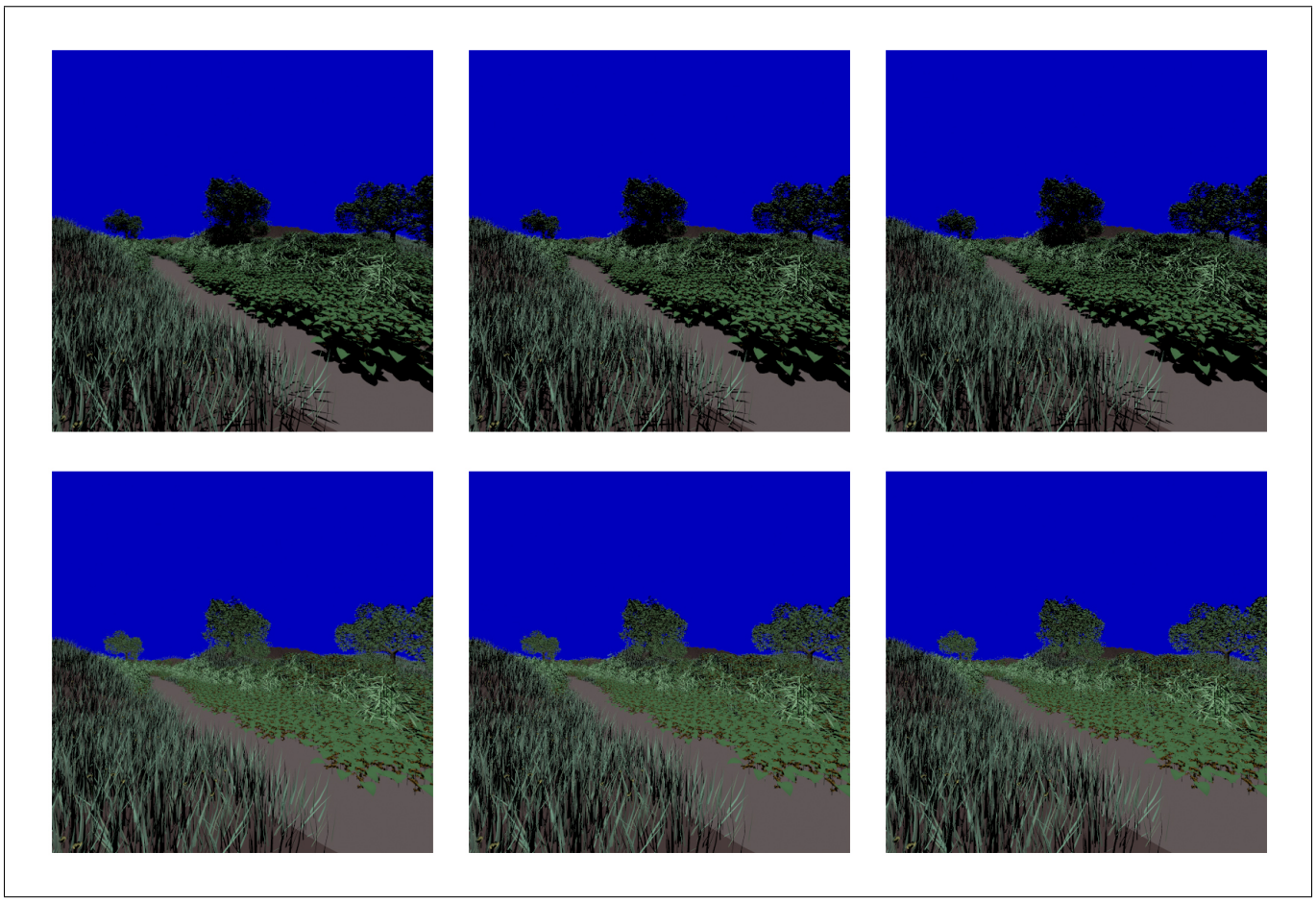

Figure 12: Single trial from experiment 1 on plant scene. From left to right: A) original 25spp image, B) K-means reduced to 8spp, C) purely sampled 8spp. Top row: With shadows. Bottom row: without shadows.

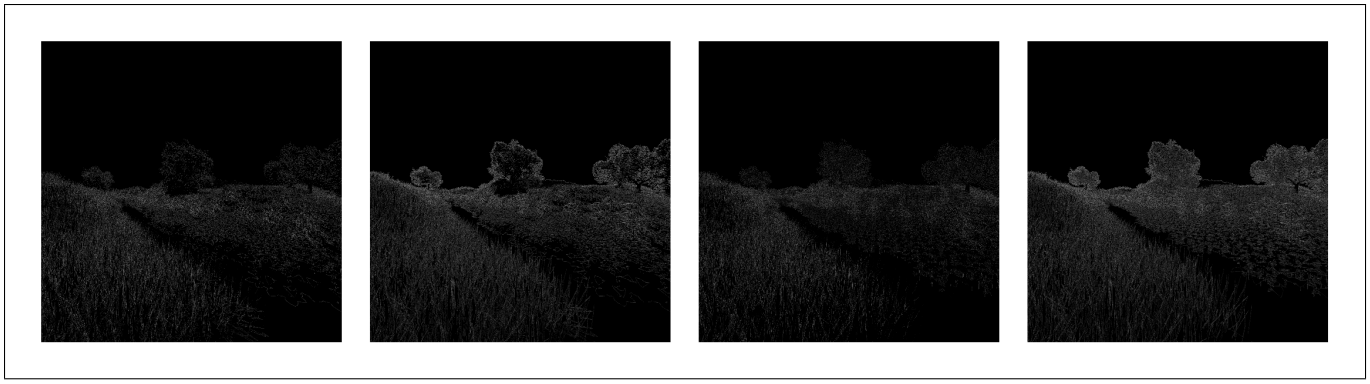

Figure 13: Images displaying error perpixel for single trial from experiment 1 on plant scene. From left to right: A) K-means reduced to 8spp with shadows, B) purely sampled 8spp with shadows, C) K-means reduced to 8spp without shadows, D) purely sampled 8spp without shadows.