Firefly removal in Monte Carlo rendering with adaptive Median of meaNs

Jérôme Buisine, Samuel Delepoulle, Christophe Renaud

Firefly removal in Monte Carlo rendering with adaptive Median of meaNs

J. Buisine, S. Delepoulle and C. Renaud

Univ. Littoral Côte d'Opale, LISIC, F-62100 Calais, France

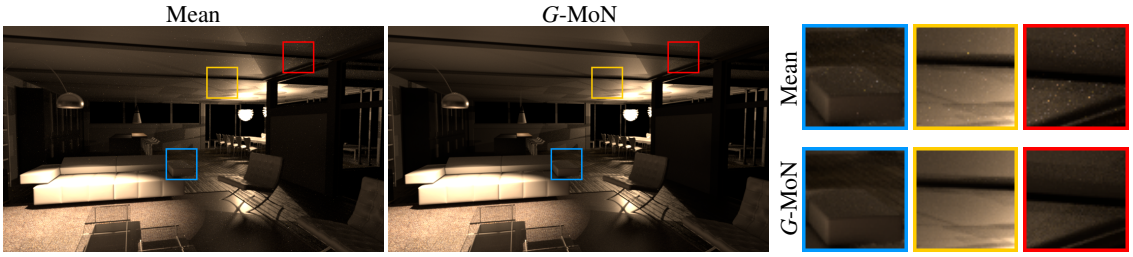


Figure 1: *Comparison of G-MoN (adaptive Median of meaNs with the use of the Gini coefficient) and classical mean estimators on Villa scene with* $100,000$ *samples using Path Tracing in both. The mean, even at* $100,000$ *samples is still sensitive to the high contributions while this estimator removes all fireflies present in the image.*

Abstract

Estimating the rendering equation using Monte Carlo methods produces photorealistic images by evaluating a large number of samples of the rendering equation per pixel. The final value for each pixel is then calculated as the average of the contribution of each sample. The mean is a good estimator, but not necessarily robust which explains the appearance of some visual artifacts such as fireflies, due to an overestimation of the value of the mean. The MoN (Median of meaNs) is a more robust estimator than the mean which allows to reduce the impact of outliers which are the cause of these fireflies. However, this method converges more slowly than the mean, which reduces its interest for pixels whose distribution does not contain outliers. To overcome this problem we propose an extension of the MoN based on the Gini coefficient in order to exploit the best of the two estimators during the computation. This approach is simple to implement whatever the integrator and does not require complex parameterization. Finally, it presents a reduced computational overhead and leads to the disappearance of fireflies.

CCS Concepts

• ***Computing methodologies*** → *Monte Carlo Techniques;* • ***Rendering*** → *Global Illumination; Sampling and Reconstruction;*

1. Introduction

Realistic image computation mimics the natural process of acquiring pictures by simulating the physical interactions of light between every existing object, light and camera lying within a 3D modelled scene. Light simulation process in a 3D scene is known as global illumination and was formalised by Kajiya [Kaj86] with the light transport rendering equation.

This equation cannot be analytically solved in most cases and Monte Carlo (MC) approaches are generally used to estimate the value of the final image pixels. Sampling is performed through the construction of random light paths between the camera and the light sources lying

in the 3D scene: in its simplest form, a ray is sent from the camera location through a pixel and is randomly reflected by the surface of the first object it encounters. The process is applied recursively until a source is encountered or the path construction is randomly stopped through the use of a Russian roulette. Numerous light paths are built per pixel according to the law of large numbers and to the MC approach. The average of the samples contribution is then computed for each pixel. It converges to the expected solution following a $1/\sqrt{n}$ rate where n is the number of samples [SWZ96]. Computing a very large number of samples per pixel makes it possible to obtain realistic synthetic images.

The final MC estimator approximation of the expected value for n samples is obtained from the empirical mean as specified in Eq. 1.

$$\bar{\theta} = \frac{1}{n}\sum_{i=0}^{n} x_i \tag{1}$$

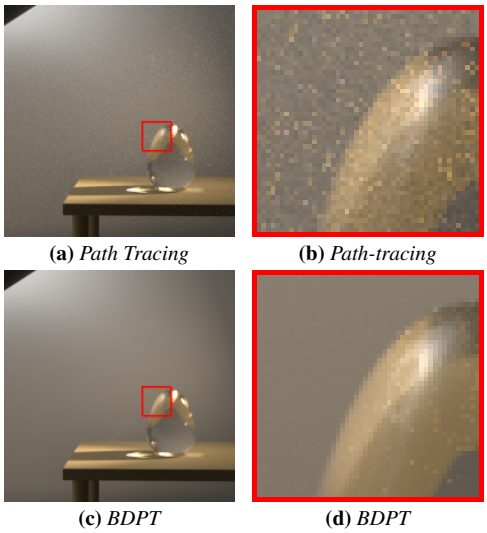
where x_i is a sample obtained during rendering.

This computation initially causes considerable noise when generating the image, but as the calculation progresses, this noise is reduced and almost invisible. When generating such images from certain scenes, however, some visual artefacts known as *fireflies* may be still present and highly perceptible to the human visual system.

Examples of such fireflies are given in Fig. 2 where two images have been generated from Veach's scene [Vea97] using Path Tracing and Bi-Directional Path Tracing (BDPT) [LW93], the latter allowing to converge more rapidly to the final image than a classical Path Tracing method for some scenes. It is quickly noticeable that many fireflies are visible when using the random path-tracing method. Although the BDPT method seems to give a better result, some fireflies are still present even with $100,000$ samples.

These artifacts mainly come from very low probability paths which contribute intensely to equation 1. Even if the mean estimator is considered as a good estimator, it is also strongly perturbed by these kind of very large values and their contribution can only be smoothed by evaluating many other samples (see Fig. 3). During the rendering of a pixel, it is difficult to decide whether the contribution of a path is such a rare value that could generate a firefly or the first occurrence of an important estimate for the pixel.

In this paper, we propose to replace the mean by more robust estimators based on the Median of meaNs (MoN), previously introduced in [JMD15]. We detail two estimators, named G-MoN and G-MoN$_b$, which use the Gini coefficient [Dor79] to identify sample distributions that are likely to exhibit firefly. The G-MoN$_b$ allows us to choose the best estimator between the mean and the MoN, while the G-MoN extends the amount of information taken into account in the MoN. Both estimators compare very favourably with the mean for firefly suppression and provide similar values to the mean when no fireflies occur. The proposed estimators have been implemented in Path Tracing, due to its greater use in production [Zhu20], but their use in other integrators is straightforward.



(a) *Path Tracing* **(b)** *Path-tracing*

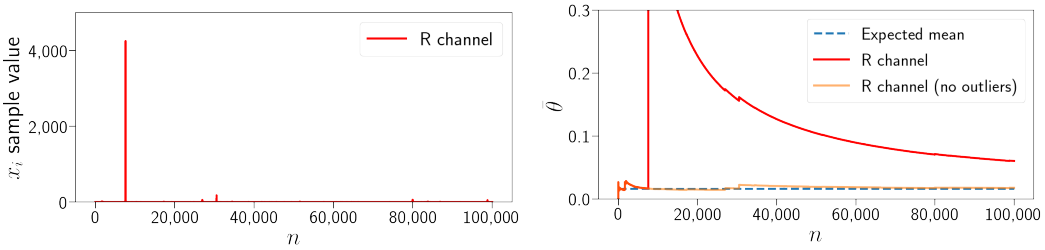
(c) *BDPT* **(d)** *BDPT*

Figure 2: *Fireflies overview of cropped part of Veach's scene using Path Tracing and BDPT with* $100,000$ *samples in both.*

The paper is organised as follows: in Section 2, the existing methods for the elimination of outliers in a statistical context are exposed as well as methods for firefly removal in rendering. Then the MoN estimator is described in Section 3 as well as its use for rendering. Our two new estimator proposals are introduced in Section 4 and their results, prior to the conclusion, are then detailed and analyzed.

2. Related work

Sample values that generate fireflies are known as *outliers* in more general scientific problems. We review here the more classical approaches that have been studied in statistic for removing such outliers as well as the approaches proposed in computer graphics.



(a) *Red spectrum luminance values of firefly pixel with 1 high contribution.*

(b) *Comparison of the mean without and with outlier value on the Red spectrum luminance values.*

Figure 3: *Impact overview of high contribution from obtained samples contribution obtained over* $\bar{\theta}$*, the arithmetic mean estimator. The y-axis of sub-figure 3b has been re-scaled to allow a better impact visualization of the high contribution values and the expected mean is computed on 10M of samples. Only the red color channel (Red spectrum) is displayed here so that highly correlated RGB spectrum values have no impact on the analysis.*

2.1. Statistical approach for outlier removing

Many indicators are available in order to well estimate the expected value in statistics, where the *filtered mean* part refers to methods where outliers are suppressed [HR09] by a clipping process. Within these methods, a few can be mentioned such as *Percentile*, *Sigma* and *Median* clipping which aim to remove certain sample values outside a confidence interval. This requires however to know the full information of the distribution of the samples. From a computer graphics perspective, it is often difficult or undesirable to have all the samples and their distribution available, due to the high memory cost and the progressive evaluation of the samples during the rendering process.

Median of meaNs (MoN) was introduced in [Bla85, JVV86] and designed to resist the appearance of so-called polluting data which are treated as outliers. It consists of separating all the samples into M sets of the same size (if possible). The mean is calculated for each of the M sets, then the median over the M sets is used as the final estimator. This estimator seems to show some robustness to values that may be outside a confidence interval [AMS99]. More recently, intensive work has been carried out on the MoN which demonstrated its robustness compared to the mean for non-normal distributions such as heavy-tail [BCBL13,BJL15,HS16]. It is now also widely studied in machine learning oriented approaches [Bü03, LM19], such as bagging and bootstrapping.

The idea of combining the robustness of the median with the consistency of the mean has been also widely studied. In [CH94] and [DP04] authors investigate the use of linear combinations of mean and median with weights picked according to asymptotic criteria.

2.2. Fireflies removal

Several works whose objective is to reduce fireflies pixels have been studied in the field of MC image rendering [DWR10, SD12]. In [DWR10] a density-based approach is proposed in order to reduce fireflies by identifying and removing them. They propose to work on a joint space of the image coordinates and the sample values (color space). Then, they use a k-d tree to iteratively store the samples which are likely to be outliers by comparing them to the k nearest neighbors samples. Although not all samples are stored in the k-d tree (only potential outlier samples), it still involves a significant storage cost. The method proposed in [SD12] aims at exploiting the relations of the samples of the neighbouring pixels to accelerate the rendering of the image while suppressing most of these fireflies.

Other approaches explore ways to better estimate the pixel value of the contributions obtained without necessarily involving additional processing, but only when generating the final image. In [JMD15], the authors have introduced median extraction on M estimators of subsampled means which appears to be identical to the MoN estimator. They propose a criterion for obtaining the value of M to be used, by evaluating the ratio between the standard deviation of the pixel and the standard deviation of the scene. If the pixel is not likely to be firefly then $M = 1$, leading to the use of the mean estimator. Otherwise, $M > 1$ (with M an odd value) with a maximum value of $M = 17$. Fireflies tend to be all removed when the number of fireflies in the scene is not too large. Furthermore the estimated value appears to be underestimated. In [ZHD18], authors specify that removing fireflies can affect the way to converge to the final solution. According to the authors, even if a sample can be iden-

tified as a firefly, it should not be removed, but weighted with respect to the progress of the calculation and distribution. They state a theoretical approach which consists in storing all the samples, sorting them, then weighting each sample according to a criterion of detection or not of an outlier. Storing all the samples is not feasible, so they propose to stratify the luminance range in a set of M buffers and to sum each sample contribution in a buffer associated with its luminance interval. Then, once sorted, each buffer B_i is assigned a reliability weight based on local and neighborhood pixel information of the sum of the samples stored in the current buffer, which allows a reduction of the outliers contribution. One of the key difference of this approach and the one proposed in this paper is the use of non-local pixel information in the case of Zirr's proposal whereas we only use the sample values local to each pixel.

Finally, some methods offer unbiased denoising approaches, i.e. ensuring that the resulting solution converges correctly. These methods are mainly oriented towards the statistical study of the samples obtained during image rendering [SD12, BB17, ETF19]. By learning how the samples are distributed, these methods aim to reduce variance and also reduce the appearance of fireflies while denoising.

The use of Deep learning techniques has also been proposed allowing both to considerably denoise the images and reducing the impact of fireflies [VRM*18, VAN*19, MH20]. They provide an expected image quality for a very low computation time, but introduce a bias.

2.3. Fireflies in Path Tracing-based integrators

Path Tracing has been improved by many other integrators, such as BDPT [LW93] or Metropolis Light Transport (MLT) [VG97] methods, mainly aiming at speeding up the convergence of the method. However, all these new methods present fireflies, to a greater or lesser degree depending on the scenes used. MLT, for example, generates mutations through a Monte Carlo Markov Chain process (MCMC) from an initial path with a significantly found contribution. It can therefore be very sensitive to the conservation of the firefly, which comes precisely from a higher contribution.

Path-guiding oriented methods have been also proposed with the aim to control the pathways that are sampled through reinforcement learning and to obtain a better knowledge of the PDF for interesting contributions [VHH*19]. Fireflies are a known aspect of practical path guiding approaches even if recent development mitigates this issue [MGN17, DGJ*20].

3. MoN in rendering

The work proposed in this paper focuses on the combination of the Gini coefficient [Dor79] and the Median of meaNs (MoN) in order to reduce the appearance of fireflies. Before detailing the two combination modes we have studied, we formally describe the MoN here. Then we detail how it can be used in the rendering and the results that can be obtained.

3.1. Definition

The MoN consists of separating all the samples obtained into M sets of the same size (if possible). The mean is calculated for each of the M sets and the the median over the M sets (the median set) is used as the final estimator. Given independent and identically distributed random sample x_i estimation, the median of means with M sets of size k with a total of $n = k \times M$ samples can be defined by the following equation:

$$\hat{\mu}_{MoN} = \text{median}\left(\frac{1}{k}\sum_{i=1}^{k} x_i, ..., \frac{1}{k}\sum_{i=n-k+1}^{n} x_i\right) \quad (2)$$

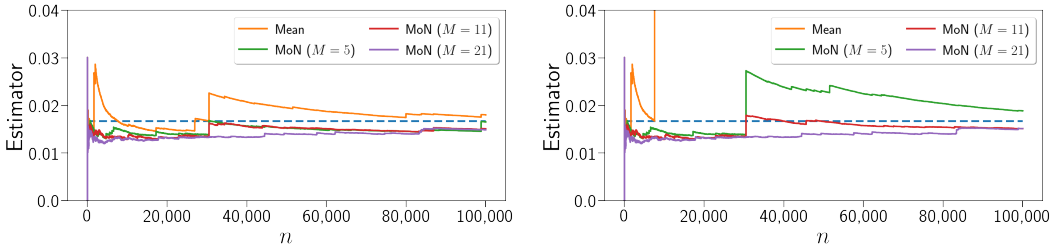
with minor adjustments if n/k is not an integer.

Implementing such an approach in a rendering engine is quite simple, but involves a storage of information that is M times larger than conventional averaging methods. However, this is still reasonable compared to saving all calculated samples. Then, during image rendering, pixels are sampled one by one, with the value of each sample added to one of the sets, for example cyclically. Finally the final pixel value is estimated through Eq. 2.

3.2. MoN behavior study

Fig. 4 allows the robustness of the MoN method to be compared to the classical mean. The number of sets used are $M = \{5, 11, 21\}$. MoN does not seem to be affected at all by the high sample contribution around the 10000th sample for each M studied (see figure 4b).

At around $30,000$ samples some contributions, higher than the previous ones are successively collected and generate a discontinuity in the convergence curve. Note that these values are not outliers since their magnitude is much smaller than that of the $10,000$th sample. Each of these contributions is stored in one of the M sets. The M means are thus impacted consequently, modifying the order of the sorted means before computing the median. This effect is clearly visible for $M = 5$ and $M = 11$ providing a little overestimate for $M = 5$. The MoN using $M = 21$ sets is not impacted in this case since the number of contributions concerned is low. These contributions are



(a) *Red spectrum luminance samples without outliers.*

(b) *Red spectrum luminance samples with outlier (in same scale).*

Figure 4: *MoN and mean comparisons over rendering samples (Red spectrum) with and without outlier. MoN is studied with $M \in \{5, 11, 21\}$ to see the impact of this parameter. The expected mean value has been computed with 10M of samples.*

thus sorted in the higher rank sets and do not modify the median set. In this case the MoN underestimates the value of the pixel.

The arithmetic mean is a good estimator, but not necessarily robust. The median, although more robust than the mean, implies a longer convergence to the expected value than the mean [M*19] and this also seems to be the case when applying the MoN, which explains this underestimation.

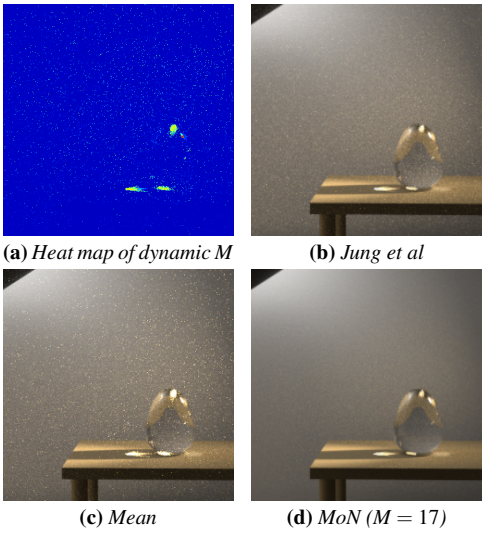
3.3. Dynamic choice of M

In [JMD15], which introduced the MoN, a dynamic choice of M was proposed in order to reduce the additional storage cost and indirectly to avoid this underestimation. The value of M is computed according to equation 3:

$$M = \begin{cases} 1 & \text{for } \frac{\sigma_{pixel}}{\sigma_{image}} < 1, \\ 2 \times \left(\lfloor \log_2 \frac{\sigma_{pixel}}{\sigma_{image}} \rfloor + 1 \right) + 1 & \text{otherwise} \end{cases} \quad (3)$$

where σ_{pixel} is the standard deviation of the outgoing radiance in a pixel and σ_{image} is the standard deviation of the overall image.

Fig. 5 provides an overview of the results obtained using equation 3. The value of M is shown as a heat map (see Sub-figure 5a) on a part of the Veach's scene, with values lying between 1 (blue) and 17 (red). It also illustrates that this dynamic M parameter tends to eliminate a large part of fireflies at $10,000$ samples (see Sub-figure 5b). However, some fireflies still remain and quite a considerable noise remains visible. On the contrary, although the MoN estimation with $M = 17$ seems to underestimate the result, it removes the fireflies and reduce the MC noise (see Sub-figure 5d). Some other results of the Jung *et al* method with the dynamic choice of M are available in Fig. 6 with SSIM scores and better overview of remained fireflies.



(a) *Heat map of dynamic M*

(b) *Jung et al*

(c) *Mean*

(d) *MoN ($M = 17$)*

Figure 5: *Jung et al dynamic M map overview of $10,000$ samples on cropped part of Veach's scene with visual results of Jung et al, mean and MoN final estimators. The heat map is composed of possible values of M ranging from 1 to 17.*

The computation of M according to equation 3 appears to have one weakness which is related to the amount of fireflies in the image: when it is high it increases the overall variance in the image. Fireflies from pixels with a lower intensity are therefore not suppressed.

The next section presents our approaches, which aim to identify the presence of firefly locally rather than globally. We also not seek to dynamically reduce the size of

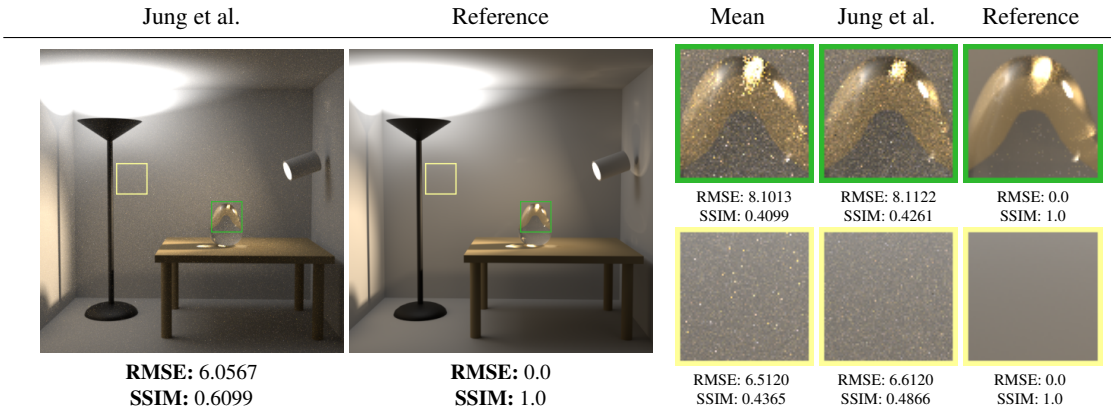


Figure 6: *Comparison of classical Mean, Jung et al with automated M and reference using BDPT images obtained with* $10,000$ *samples per pixel.*

M, but instead extend the amount of information taken into account in the MoN.

4. Adaptive MoN with the Gini coefficient

In this section we propose to improve the MoN estimator that is helpful in reducing the impact of outliers and thus the appearance of fireflies. The two approaches will be described that will focus in identifying fireflies locally (for each pixel independently). They are both based on the use of the Gini coefficient but in different ways.

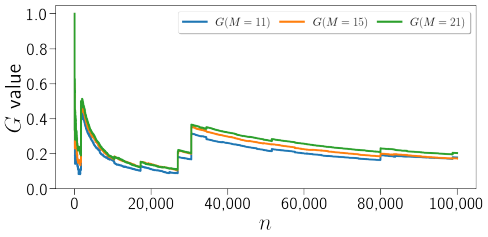
4.1. Firefly detection

The Gini coefficient [Dor79] is used in econometrics to highlight social inequalities. If the value obtained from this coefficient is 0 then there is a perfect equality and 1 (which cannot be achieved) means total inequality. In this paper, we focus on this coefficient in order to detect the presence or not of a firefly with the idea that adding the value of a firefly to one of the MoN means should increase the inequality between these means.

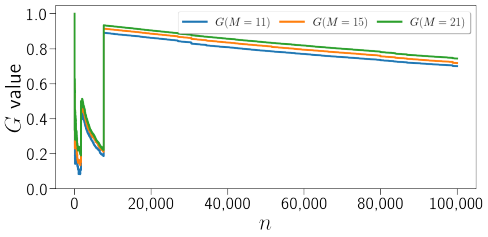
The Gini coefficient applied to the MoN can be calculated by equation 4, based on a reformulation of the equation proposed in [DW00]. Our reformulation is detailed in appendix A.

$$G = \frac{2\sum_{j=1}^{M} j\hat{\theta}_j}{M\sum_{j=1}^{M}\hat{\theta}_j} - \frac{M+1}{M} \tag{4}$$

with $\hat{\theta}_j$, $j \in [1,M]$, the M means computed with the MoN, indexed in ascending order ($\hat{\theta}_j \leq \hat{\theta}_{j+1}$).



(a) *G coefficient evolution on samples with no outliers.*



(b) *G coefficient evolution on samples with some outliers.*

Figure 7: *Gini coefficient evolution over M-means obtained during rendering with* $M \in \{11,15,21\}$ *to see the impact of outliers over G.*

We show in Fig. 7 the evolution of the G coefficient for the same sample distributions that appear in Fig. 4 with the firefly that appears around the $10,000$th sample removed in Sub-figure 7a. This sub-figure clearly indicates that G seems to trust the obtained averages because large outliers are not present here. If a slightly larger sample value appears (for example around $n = 30,000$), the coefficient fluctuates moderately but still specifies that the distribution of the means is not uneven, since

$G \leq 0.4$. On the contrary, in sub-figure 7b, G coefficient reacts strongly when it encounters the outlier around $n = 10,000$, reaching a value close to 1, signifying the appearance of an inequality between the means. Even with $100,000$ samples the decrease of the value is not sufficient for removing the lack of confidence in the equality of the obtained M means. Let us note that the larger the value of M, the more difficult it appears to achieve equality since inequality with respect to the other sets is common and influence the G coefficient.

One of the advantages of using the Gini coefficient is that it adjusts if the inequality decreases. In the context of our approach, this would mean that a value identified as an outlier has been detected causing high inequality in the M sets. If, however, some other sets also get same magnitude contributions, the value of the Gini coefficient will reduce, trusting again in the equality of the different means and considering that these values are meaningful ones.

4.2. Binary Median of meaNs

Because of the behaviour of G according to the presence or absence of fireflies, we define a threshold below which we consider that full confidence can be given to the estimate provided by the classical mean. For example, a value of $G \leq 0.25$ will correspond to a case in which the confidence indicator for the mean is risk-free (without outlier and therefore without firefly).

We thus propose a new estimator based on this confidence criterion, so called G-MoN$_b$, defined by:

$$G\text{-MoN}_b = \begin{cases} \bar{\theta} & \text{if } G \leq 0.25, \\ \hat{\mu}_{MoN} & \text{otherwise} \end{cases} \tag{5}$$

This estimator will exploit the G coefficient in order to define whether it should be oriented towards the mean or towards the MoN estimator. It is denoted G-MoN$_b$, for binary, as it makes a purely binary choice (mean or MoN).

4.3. Adaptive Median of meaNs

As mentioned in [Ore19], in spite of its theoretical properties, the MoN under utilizes the data available by only using the median set. Based on this idea, this adaptive MoN approach wishes to take advantage of the information available in the neighboring sets of the median set. It will be defined by the following equation:

$$\hat{\mu}_{G\text{-}MoN} = \frac{\sum_{j=1+c}^{M-c} \hat{\theta}_j}{M - 2c} \tag{6}$$

with $\hat{\theta}_j$, $j \in [1, M]$, the M means computed with the MoN, indexed in ascending order ($\hat{\theta}_j \leq \hat{\theta}_{j+1}$) and $c \in \mathbb{N}^{[0, \lfloor \frac{M}{2} \rfloor]}$ the number of means from each extreme side of the M ranked means that will not be taken into account for the final estimator.

Because of the ability of the Gini coefficient to provide some information about the equality/inequality between a set of information, we use it for computing dynamically the value of the c parameter introduced in equation 6 such as: $c = \lfloor G \times k \rfloor$ with $k = \lfloor \frac{M}{2} \rfloor$. Thus, if G gives us a strong tie, then the final estimator will be close to the mean by using more neighbouring means. Otherwise, it will be close to the MoN estimator by using only the median set, but with potentially some additional information from neighbouring sets.

5. Comparisons and Results

In this section we present the results of these two estimators during rendering for different images and compare them to three other estimators: the mean, the MoN and the Jung *et al*'s version of the MoN.

5.1. Experimentation setup

Our experiments were conducted using the Physically Based rendering engine *PBRT-v4* [PJH16] with GPU support and available literature scenes [Bit16]. In the context of computer-generated images, $100,000$ samples remains a consequent large number of samples, but does not seem to be able to erase the presence of firefly for each scene. In order to check which estimators seem to be more interesting than others when rendering, reference images of 4 scenes have been computed and are available in Fig. 8. The Veach and Villa scenes which references are available in sub-figures 8a and 8d are highly sensitive to fireflies, unlike Bathroom and Crown scenes in sub-figures 8b and 8c that do not provide any firefly. The interest is to well compare the robustness of each estimator according to the nature of the scene.

When comparing MoN based estimators on different M values, such as classical MoN, G-MoN$_b$ and G-MoN, Jung *et al*'s method is ran using its dynamic M parameter with recommended $M = 17$ as the maximum possible number of sets. The mean remains fixed (not impacted by the choice of M parameter).

The Zirr *et al*'s approach [ZHD18] was also implemented in PBRT for comparisons. This method uses M buffers with ordered luminance intervals that are weighted for final estimation. The computed weight is

(a) *Veach* **(b)** *Bathroom*

(c) *Crown* **(d)** *Villa*

Figure 8: *Reference images used for comparisons of estimators: Veach and Villa are computed with* $100,000$ *samples using BDPT, Bathroom and Crown with 1M samples using Path Tracing.*

relative to a reliability of the current pixel and its neighbors (3×3 kernel size), which differentiates it from the previously compared estimators where only local information are used, even if the approach remains very similar. They proposed a parameter b allowing to define the size of each luminance interval where the interval of buffer B_j is in range $[b^{j-1}, b^{j+1}]$. The b parameter can be set automatically with $b = \sqrt[M]{S_{max}}$ where S_{max} is the maximum expected luminance set as 8^6 such as in the Zirr *et al*'s paper.

5.2. Convergence study

A convergence study of the different proposed estimators (i.e. G-MoN and G-MoN$_b$) is processed in comparison with the Jung *et al* (with automated M), MoN and mean. The results are available in Fig. 9 using the structural similarity index measure (SSIM) more correlated with the Human Visual System (HVS) based on the idea that the pixels have strong interdependencies especially when they are spatially close. The classical Root-Mean-Square Error (RMSE) was only indicated since its sensitivity to firefly is low: it calculates an absolute error, whereas a firefly generates a very local error as compared to the reference image.

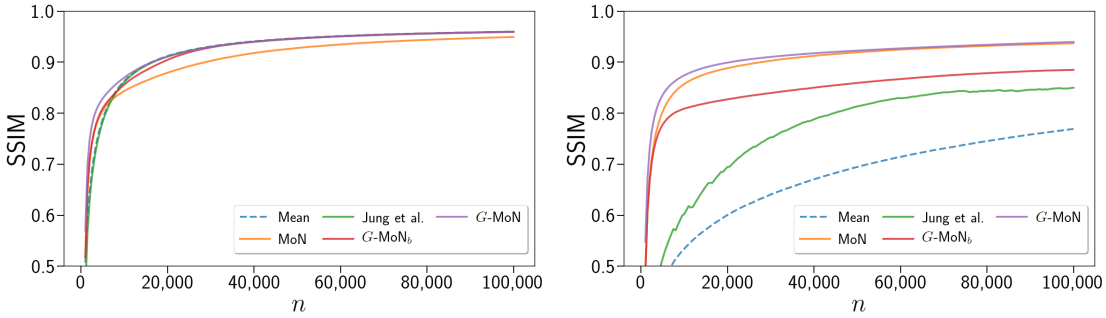
Sub-figure 9a presents results for the scene Bathroom that does not highlight any firefly. All the estimators have a very close convergence rate, with a slightly lower rate for MoN which is known for underestimating its results. None of the estimators is therefore penalised by the absence of firefly. When images highlight fireflies, such as in the Veach's scene, the difference between the different estimators becomes more important (see sub-figure 9b). The mean as well as Jung *et al*'s method fail due to the presence of a large number of fireflies that are perceived by SSIM. The way the parameter M is computed in Jung *et al*'s approach is sensitive to such situations and reduce its efficiency. G-MoN and MoN provide the best results in removing fireflies, with a slight advantage to G-MoN. Even if the two estimators finally have a similar convergence when fireflies are visible, G-MoN is more robust since providing better results in images without fireflies. G-MoN$_b$ shows a weakness in robustness, its SSIM score converging between those of the mean and MoN, but remaining worse than those obtained by MoN or G-MoN.

To complete the convergence study results, table 1 presents the SSIM scores obtained by the estimators G-MoN, G-MoN$_b$ and MoN for $M = \{5, 11, 15, 21, 25\}$ for the 4 scenes computed with $100,000$ samples and Path Tracing. These estimators are compared to Jung *et al* and to the mean which SSIM score values are indicated again for each M value (the mean is not dependent on M and we have used a constant maximal value of $M = 17$ for Jung). By studying the proposed rank based on SSIM scores, G-MoN seems to be better each time on at least 3 scenes when $M > 11$. The only scene where it seems to be a bit further away is the Bathroom scene where its SSIM score remains very good which confirms that it is a consistent and reliable estimator.

Additional results on the convergence of each estimator are also provided in the table 2. The number of samples required is given in relation to a SSIM score achieved for each of the 4 scenes. Proposed SSIM scores are $\in \{0.6, 0.7, 0.8\}$. It can be quickly noticed that the G-MoN estimator almost always reaches first the SSIM fixed score in fewer samples than all other estimators, especially when $M > 5$. An interpretation of such results, is that with a large enough value of M, i.e. $M \in [11, 25]$, the Gini coefficient seems to capture more information from the M means and allows a better final estimate.

5.3. Visual impact of estimators

We also visually compare the results obtained for the two estimators with respect to Jung *et al*, mean and the classical MoN. For this purpose, we compare images computed with $10,000$ samples to the reference ones. Fig. 10 presents such comparisons for the two images Veach and Bathroom. Two targeted areas are indicated by a respective colored dial in the full image.



(a) *Comparisons of 5 estimators using SSIM during rendering on the Bathroom image.*

(b) *Comparisons of 5 estimators using SSIM during rendering of the Veach's scene image.*

Figure 9: *Convergence study of G based estimators with* $M = 21$ *using SSIM indicator until* $100,000$ *samples. Estimators are compared to Jung et al, classical MoN (with M = 21) and mean on two scenes using the reference images available in Fig. 8: Bathroom, with no fireflies (left) and Veach with a large amount of fireflies (right).*

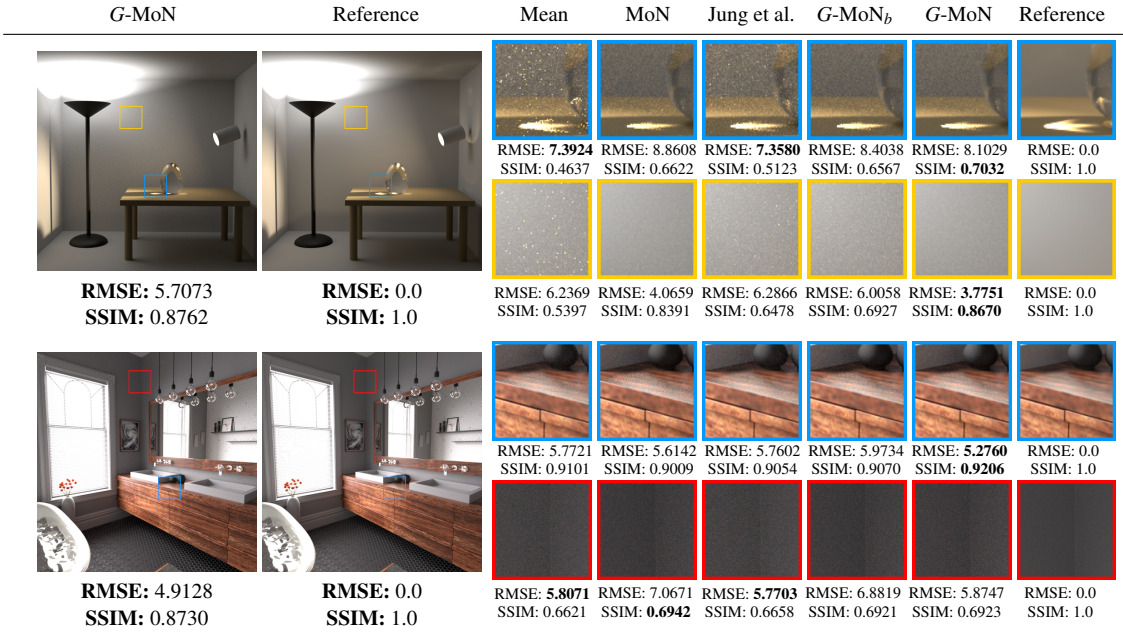


Figure 10: *Comparisons of RMSE and SSIM obtained from different estimators with* $10,000$ *samples on 2 images. Full size images and targeted areas are compared to references. MoN, G-MoN$_b$ and G-MoN are set with* $M = 21$.

For the Veach's scene image that highlights numerous fireflies, the MoN based approaches provide the best visual results with the exception of the Jung's approach which global criterion keeps several fireflies. In terms of SSIM score, *G*-MoN seems closest to the reference. Adding knowledge from neighbouring sets of the median set brings a better precision of estimation.

For the Bathroom image, all estimators seem to be relatively close to the reference either visually or in SSIM score. However, MoN seems to be more accurate in uniform areas that are not sensitive to fireflies (see the red area in the figure). *G*-MoN, on the other hand, proposes a score faithful to the reference on more complex areas or very close to the best estimator (such as MoN for the

	Scene	Veach	Bathroom	Crown	Villa
$M = 5$	Mean	0.77061 (5)	0.95987 (2)	0.99377 (2)	0.86733 (5)
	MoN	**0.88315** (1)	0.95275 (5)	0.99296 (4)	0.89738 (2)
	Jung et al.	0.84992 (3)	0.95932 (3)	0.99263 (5)	0.87744 (4)
	G-MoN$_b$	0.84036 (4)	**0.95988** (1)	**0.99378** (1)	0.89493 (3)
	G-MoN	0.88017 (2)	0.95785 (4)	0.99363 (3)	**0.89858** (1)
$M = 11$	Mean	0.77061 (5)	0.95987 (2)	0.99377 (3)	0.86733 (5)
	MoN	0.91602 (2)	0.95074 (5)	0.99280 (4)	**0.90833** (1)
	Jung et al.	0.84992 (4)	0.95932 (4)	0.99263 (5)	0.87744 (4)
	G-MoN$_b$	0.87084 (3)	**0.95993** (1)	0.99378 (2)	0.90461 (2)
	G-MoN	**0.92090** (1)	0.95935 (3)	**0.99380** (1)	0.89744 (3)
$M = 15$	Mean	0.77061 (5)	**0.95987** (1)	0.99377 (2)	0.86733 (5)
	MoN	0.92697 (2)	0.95003 (5)	0.99279 (4)	0.91249 (2)
	Jung et al.	0.84992 (4)	0.95932 (4)	0.99263 (5)	0.87744 (4)
	G-MoN$_b$	0.87818 (3)	0.95979 (2)	0.99377 (3)	0.90814 (3)
	G-MoN	**0.93097** (1)	0.95950 (3)	**0.99383** (1)	**0.91329** (1)
$M = 21$	Mean	0.77061 (5)	**0.95987** (1)	0.99377 (2)	0.86733 (5)
	MoN	0.93711 (2)	0.94953 (5)	0.99273 (4)	0.91681 (2)
	Jung et al.	0.84992 (4)	0.95932 (4)	0.99263 (5)	0.87744 (4)
	G-MoN$_b$	0.88508 (3)	0.95978 (2)	0.99374 (3)	0.91190 (3)
	G-MoN	**0.93976** (1)	0.95965 (3)	**0.99383** (1)	**0.91739** (1)
$M = 25$	Mean	0.77061 (5)	**0.95987** (1)	0.99377 (2)	0.86733 (5)
	MoN	0.94146 (2)	0.94915 (5)	0.99278 (4)	0.91894 (2)
	Jung et al.	0.84992 (4)	0.95932 (4)	0.99263 (5)	0.87744 (4)
	G-MoN$_b$	0.88769 (3)	0.95967 (2)	0.99377 (3)	0.91389 (3)
	G-MoN	**0.94359** (1)	0.95953 (3)	**0.99387** (1)	**0.91952** (1)

Table 1: *SSIM comparison with reference image and different M values using $100,000$ samples with Path Tracing. The rank of the studied estimators is from (1) to (5). The mean and Jung et al estimators are added for each different M row for comparison purposes.*

red boxed area). In addition, G-MoN seems to converge faster than the arithmetic mean. The benefit of filtering values far from the mean by applying the median brings better results.

All the experiments and results obtained seem to indicate that the G-MoN estimator gives a good estimate. It removes fireflies while being close to or above the classical mean for non-firefly scenes. The dynamic choice of the c parameter set number using the G coefficient seems to be robust enough. G-MoN$_b$ on the other hand, also allow the suppression of the fireflies, but is still not as good as the MoN, especially on the Veach's scene. G-MoN brings robustness and fidelity of the desired estimator.

Rather than trying to automate the M parameter, exploiting the separation of the samples into sets allows us to identify fireflies and improve the reliability of the final estimator.

5.4. Non-local comparison

We compare our approach to the Zirr's method that appears to be globally better than the G-MoN method. The convergence curves presented in figure 11, seem to indicate that even with a small value of M (e.g. $M = 5$), the estimator converges more quickly. For $M = 25$, G-MoN and MoN seem to converge, but still perform less well than Zirr's method. This is mainly due to the use of neighbouring information for the computation of sample weights (i.e. luminance interval buffers) which has not been exploited in our approach.

Figure 12 illustrates the images obtained for the G-MoN and Zirr *et al* methods where a visual proximity of the two images can be seen.

The complexity of Zirr's method based on a weighting theoretical approach allows it to be robust to fireflies, but it should be noted that the G-MoN one, even if it performs less efficiently on smaller sizes of M, is still competitive and easier to implement because of its simplicity.

5.5. Computation overhead

Fig. 1 provides a rendering overview of the Villa scene for the G-MoN and the mean estimators. Both images were computed on the GPU using Path Tracing with $100,000$ samples per pixel. Visually, all the fireflies are removed by using G-MoN. The computation times are 3618.3s and 3576.3s for G-MoN and the classical mean respectively. Thus G-MoN estimator only leads to an additional cost of $\approx 1.17\%$ for the management of the MoN sets and the estimation of the final pixel value.

MoN-based integrators require rendering to M different sets, each one representing a mean. This involves an additional memory cost, that stays however relatively low. Each set has to store a partial sum and the number of samples included in the sum, thus 8 bytes if computations are performed in single precision. Consequently, if pixels are computed independently only $8 \times M$ additional bytes are required for each channel of the spectrum. When filtering is used, several pixels have to be stored simultaneously. However, parallel computation often operates on a pixel-patch basis and only the memory needed for the path size is required. Assuming 32×32 sized patches, (R, G, B) spectrum and $M = 21$, this will require less than 500 kB of additional memory per patch.

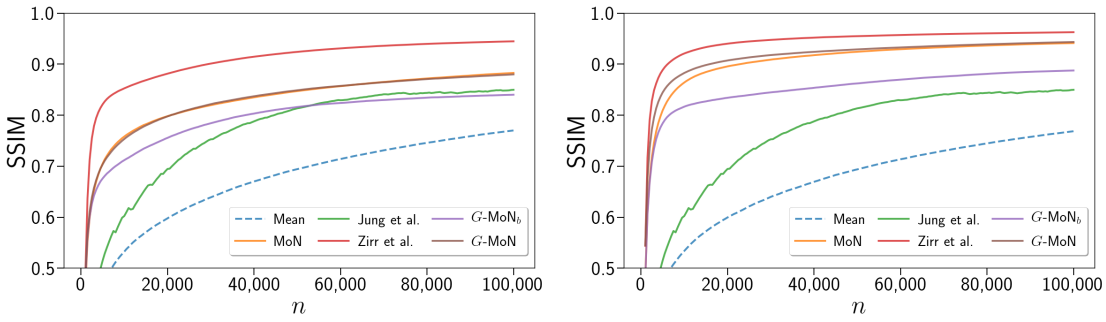
Let us note that in case of progressive rendering the buffers of all the pixels have to be saved in order to allow the integrator to continue the computation. This is thus similar to store M image buffers. The buffers used in the Zirr's approach must also be stored, with the advantage that their number is smaller.

6. Conclusion

In this paper we have focused on the fireflies problem, artefacts that reduce the quality of images produced by lighting simulations. We have recalled the problem of the mean estimator when outliers values appear and review the Median of meaNs as a valuable alternative to this classical estimator. The Gini coefficient is introduced, in order to investigate the presence of outlier values locally for each pixel. Then two new estimators are proposed by

Scene		Veach			Bathroom			Crown			Villa		
SSIM		0.6	0.7	0.8	0.6	0.7	0.8	0.6	0.7	0.8	0.6	0.7	0.8
$M=5$	Mean	20007 (5)	52592 (5)	NR (5)	1074 (4)	2236 (3)	**5021** (1)	244 (4)	308 (4)	392 (4)	498 (4)	2507 (5)	14515 (5)
	MoN	1518 (3)	**4256** (1)	**20353** (1)	729 (3)	2846 (5)	7501 (5)	239 (2)	302 (2)	383 (2)	343 (3)	469 (3)	1519 (3)
	Jung et al.	9163 (4)	20644 (4)	44387 (4)	1261 (5)	2430 (4)	5233 (3)	265 (5)	337 (5)	435 (5)	525 (5)	2412 (4)	12310 (4)
	G-MoN$_b$	1491 (2)	7515 (3)	37626 (3)	646 (2)	2180 (2)	5152 (2)	241 (3)	304 (3)	387 (3)	340 (2)	463 (2)	1422 (2)
	G-MoN	**1378** (1)	4300 (2)	20394 (2)	**627** (1)	**2163** (1)	5547 (4)	**233** (1)	**293** (1)	**370** (1)	**332** (1)	**448** (1)	**1368** (1)
$M=11$	Mean	20007 (5)	52592 (5)	NR (5)	1074 (4)	2236 (4)	5021 (2)	244 (4)	308 (4)	392 (4)	498 (4)	2507 (5)	14515 (5)
	MoN	1060 (3)	2222 (2)	6835 (2)	619 (3)	1641 (3)	7066 (5)	240 (3)	303 (3)	384 (3)	364 (3)	503 (3)	1395 (3)
	Jung et al.	9163 (4)	20644 (4)	44387 (4)	1261 (5)	2430 (5)	5233 (4)	265 (5)	337 (5)	435 (5)	525 (5)	2412 (4)	12310 (4)
	G-MoN$_b$	1031 (2)	2411 (3)	19398 (3)	607 (2)	1532 (2)	5198 (3)	239 (2)	302 (2)	383 (2)	363 (2)	500 (2)	1378 (2)
	G-MoN	**671** (1)	**1566** (1)	**5243** (1)	**508** (1)	**1254** (1)	**4537** (1)	**229** (1)	**288** (1)	**362** (1)	**344** (1)	**465** (1)	**1126** (1)
$M=15$	Mean	20007 (5)	52592 (5)	NR (5)	1074 (4)	2236 (4)	5021 (3)	244 (4)	308 (4)	392 (4)	498 (4)	2507 (5)	14515 (5)
	MoN	986 (3)	1977 (2)	5224 (2)	649 (3)	1474 (3)	5636 (5)	242 (3)	305 (3)	387 (3)	392 (3)	551 (3)	1448 (3)
	Jung et al.	9163 (4)	20644 (4)	44387 (4)	1261 (5)	2430 (5)	5233 (4)	265 (5)	337 (5)	435 (5)	525 (5)	2412 (4)	12310 (4)
	G-MoN$_b$	983 (2)	2100 (3)	12702 (3)	642 (2)	1447 (2)	4734 (2)	240 (2)	303 (2)	386 (2)	390 (2)	549 (2)	1444 (2)
	G-MoN	**634** (1)	**1361** (1)	**3676** (1)	**520** (1)	**1061** (1)	**3706** (1)	**230** (1)	**288** (1)	**362** (1)	**366** (1)	**501** (1)	**1199** (1)
$M=21$	Mean	20007 (5)	52592 (5)	NR (5)	1074 (4)	2236 (4)	5021 (4)	244 (3)	308 (3)	392 (3)	498 (4)	2507 (5)	14515 (5)
	MoN	941 (2)	1819 (2)	4377 (2)	719 (3)	1416 (2)	4179 (3)	245 (4)	309 (4)	393 (4)	442 (3)	652 (3)	1593 (2)
	Jung et al.	9163 (4)	20644 (4)	44387 (4)	1261 (5)	2430 (5)	5233 (5)	265 (5)	337 (5)	435 (5)	525 (5)	2412 (4)	12310 (4)
	G-MoN$_b$	964 (3)	1927 (3)	7142 (3)	716 (2)	1418 (3)	3961 (2)	243 (2)	307 (2)	391 (2)	441 (2)	650 (2)	1596 (3)
	G-MoN	**625** (1)	**1236** (1)	**2829** (1)	**567** (1)	**1017** (1)	**2766** (1)	**231** (1)	**290** (1)	**365** (1)	**411** (1)	**580** (1)	**1342** (1)
$M=25$	Mean	20007 (5)	52592 (5)	NR (5)	1074 (4)	2236 (4)	5021 (4)	244 (2)	308 (2)	392 (2)	498 (4)	2507 (5)	14515 (5)
	MoN	932 (2)	1780 (2)	4112 (2)	774 (3)	1422 (2)	3804 (3)	246 (4)	311 (4)	395 (4)	479 (3)	736 (3)	1727 (2)
	Jung et al.	9163 (4)	20644 (4)	44387 (4)	1261 (5)	2430 (5)	5233 (5)	265 (5)	337 (5)	435 (5)	525 (5)	2412 (4)	12310 (4)
	G-MoN$_b$	962 (3)	1886 (3)	6025 (3)	772 (2)	1428 (3)	3742 (2)	244 (2)	309 (3)	394 (3)	477 (2)	734 (2)	1732 (3)
	G-MoN	**631** (1)	**1199** (1)	**2626** (1)	**609** (1)	**1040** (1)	**2517** (1)	**232** (1)	**291** (1)	**366** (1)	**443** (1)	**644** (1)	**1446** (1)

Table 2: *Required samples per pixel in order to reach a SSIM score compared to reference images with different M values. The maximum number of samples is fixed at $100,000$, NR, for Not Reached is indicated when SSIM score cannot be reached. Mean and Jung et al are added for each different M row for comparison purposes.*



(a) *Comparisons of Zirr and previous studied estimators using SSIM and $M=5$ on Veach scene.*

(b) *Comparisons of Zirr and previous studied estimators using SSIM and $M=25$ on Veach's scene.*

Figure 11: *Convergence study of Zirr's method and previous studied estimators with $M \in \{5, 25\}$ using SSIM indicator until $100,000$ samples. Estimators are compared on Veach's scene with a large amount of fireflies.*

combining the MoN with this coefficient with so as to increasing the robustness of the MoN.

These estimators use only pixel internal information and their comparisons with similar estimators previously proposed showed that G-MoN was robust enough to remove fireflies fairly quickly. G-MoN implementation within any integrator is simple and its additional processing cost remains low in comparison with the proposed results. The method is furthermore very simple to parallelize as it does not require any global information, but only local information from the pixel itself. The comparison we made with the Zirr's method highlights that this last one provides a better convergence than the proposed method. But it requires neighborhood information in order to compute buffers weight with more reliability. When M is large enough, G-MoN seems to be quite close to Zirr *et al* but with only local information.

Future works should focus on some other recent estimators based on the MoN, such as the Permutation Invariant Median of Means Estimator [Poe21], the Tournament

Figure 12: *Comparisons of G-MoN and Zirr et al methods with* $M = 25$ *after* $100,000$ *samples on Veach's scene.*

MoN [LM16] or the Bayesian MoN [Ore19], in order to determine whether they could have any interest in fireflies removal or noise reduction. Then, we plan to more deeply study the choice of the best value for the Gini Coefficient. We have experimentally found that $G = 0.25$ appears to provide good results globally on the images. However, it could be relevant to study whether its value could be adjusted more locally according to the sample distributions that can be very different between the pixels. As illustrated by Zirr's method neighborhood information is able to improve the results of the estimators. Thus we will study the ways to use such neighbor pixel information (buffer data, Gini coefficients, ...) in order to improve the convergence rate of *G*-MoN.

The behaviour of these methods should also be studied and compared on sequences and animations to check whether temporal artefacts can be visible.

Finally, it could be interesting to investigate the use of Gini coefficient for noise detection and/or denoising, due to its ability to judge the equality in a set of data.

7. Acknowledgements

The authors would like to thank the anonymous reviewers for their insightful comments and advice which helped to improve this article.

This research was funded by ANR support: project ANR-17-CE38-0009. Experiments presented in this paper were carried out using the CALCULCO computing platform, supported by SCoSI/ULCO (Service COmmun du Système d'Information de l'Université du Littoral Côte d'Opale)

Appendix A: Gini coefficient formulation

The formulation of the Gini coefficient G used in equation 4 has been easily derived from the formulation provided in [DW00] and [Dam03], when data are ordered by increasing size of values. G is given as $G = \frac{\sum_{i=1}^{n}(2i-n-1)x_i}{n^2\mu}$, with $\mu = \frac{\sum_{i=1}^{n} x_i}{n}$. Then:

$$\begin{aligned} G &= 2\frac{\sum_{i=1}^{n} i.x_i}{n^2\mu} - \frac{(n+1)\sum_{i=1}^{n} x_i}{n^2\mu} \\ &= 2\frac{\sum_{i=1}^{n} i.x_i}{n^2\frac{1}{n}\sum_{i=1}^{n} x_i} - (n+1)\frac{1}{n}\sum_{i=1}^{n} x_i \times \frac{1}{n\mu} \\ &= \frac{2\sum_{i=1}^{n} i.x_i}{n\sum_{i=1}^{n} x_i} - (n+1)\frac{\mu}{n\mu} \\ &= \frac{2\sum_{i=1}^{n} i.x_i}{n\sum_{i=1}^{n} x_i} - \frac{n+1}{n} \end{aligned} \tag{7}$$

References

[AMS99] ALON N., MATIAS Y., SZEGEDY M.: The Space Complexity of Approximating the Frequency Moments. Journal of Computer and System Sciences 58, 1 (Feb. 1999), 137–147. 3

[BB17] BOUGHIDA M., BOUBEKEUR T.: Bayesian collaborative denoising for monte carlo rendering. Computer Graphics Forum (Proc. EGSR 2017) 36, 4 (2017), 137–153. 4

[BCBL13] BUBECK S., CESA-BIANCHI N., LUGOSI G.: Bandits With Heavy Tail. IEEE Transactions on Information Theory 59, 11 (Nov. 2013), 7711–7717. 3

[Bit16] BITTERLI B.: Rendering resources, 2016. https://benedikt-bitterli.me/resources/. 7

[BJL15] BROWNLEES C., JOLY E., LUGOSI G.: Empirical risk minimization for heavy-tailed losses. The Annals of Statistics 43, 6 (Dec. 2015). 3

[Bla85] BLAIR C.: Problem Complexity and Method Efficiency in Optimization (A. S. Nemirovsky and D. B. Yudin). SIAM Review 27, 2 (June 1985), 264–265. 3

[Bü03] BÜHLMANN P.: Bagging, Subagging and Bragging for Improving some Prediction Algorithms. In Recent Advances and Trends in Nonparametric Statistics. Elsevier, 2003, pp. 19–34. 3

[CH94] CHAN Y. M., HE X.: A simple and competitive estimator of location. Statistics & Probability Letters 19, 2 (Jan. 1994), 137–142. 3

[Dam03] DAMGAARD C.: "gini coefficient." from mathworld–a wolfram web resource, 2003. URL: https://mathworld.wolfram.com/GiniCoefficient.html. 12

[DGJ*20] DIOLATZIS S., GRUSON A., JAKOB W., NOWROUZEZAHRAI D., DRETTAKIS G.: Practical Product Path Guiding Using Linearly Transformed Cosines. Computer Graphics Forum 39, 4 (2020), 23–33. 4

[Dor79] DORFMAN R.: A formula for the gini coefficient. The review of economics and statistics (1979), 146–149. 2, 4, 6

[DP04] DAMILANO G., PUIG P.: Efficiency of a Linear Combination of the Median and the Sample Mean: The Double Truncated Normal Distribution. Scandinavian Journal of Statistics 31, 4 (2004), 629–637. 3

[DW00] DAMGAARD C., WEINER J.: Describing inequality in plant size or fecundity. 1139–1142. doi:https://doi.org/10.1890/0012-9658(2000)081[1139:DIIPSO]2.0.CO;2. 6, 12

[DWR10] DECORO C., WEYRICH T., RUSINKIEWICZ S.: Density-based outlier rejection in monte carlo rendering. In Computer Graphics Forum (2010), vol. 29, Wiley Online Library, pp. 2119–2125. 3

[ETF19] ELEK O., THOMAS M. M., FORBES A.: Learning patterns in sample distributions for monte carlo variance reduction. arXiv preprint arXiv:1906.00124 (2019). 4

[HR09] HUBER P. J., RONCHETTI E.: Robust statistics. 2nd john wiley & sons. Hoboken, NJ (2009). 3

[HS16] HSU D., SABATO S.: Loss minimization and parameter estimation with heavy tails. arXiv:1307.1827 [cs, stat] (Apr. 2016). arXiv: 1307.1827 version: 7. 3

[JMD15] JUNG J. W., MEYER G., DELONG R.: Robust statistical pixel estimation. In Computer Graphics Forum (2015), vol. 34, Wiley Online Library, pp. 585–596. 2, 3, 5

[JVV86] JERRUM M. R., VALIANT L. G., VAZIRANI V. V.: Random generation of combinatorial structures from a uniform distribution. Theoretical Computer Science 43 (1986), 169–188. 3

[Kaj86] KAJIYA J. T.: The rendering equation. In Proceedings of the 13th annual conference on Computer graphics and interactive techniques (1986), pp. 143–150. 1

[LM16] LUGOSI G., MENDELSON S.: Risk minimization by median-of-means tournaments. arXiv preprint arXiv:1608.00757 (2016). 12

[LM19] LUGOSI G., MENDELSON S.: Risk minimization by median-of-means tournaments, Dec. 2019. 3

[LW93] LAFORTUNE E. P., WILLEMS Y. D.: Bi-directional path tracing. 2, 4

[M*19] MINSKER S., ET AL.: Distributed statistical estimation and rates of convergence in normal approximation. Electronic Journal of Statistics 13, 2 (2019), 5213–5252. 5

[MGN17] MÜLLER T., GROSS M., NOVÁK J.: Practical Path Guiding for Efficient Light-Transport Simulation. Computer Graphics Forum 36, 4 (2017), 91–100. 4

[MH20] MUNKBERG J., HASSELGREN J.: Neural denoising with layer embeddings. 1–12. URL: https://onlinelibrary.wiley.com/doi/abs/10.1111/cgf.14049, doi:https://doi.org/10.1111/cgf.14049. 4

[Ore19] ORENSTEIN P.: Robust Mean Estimation with the Bayesian Median of Means. arXiv:1906.01204 [math, stat] (June 2019). arXiv: 1906.01204. 7, 12

[PJH16] PHARR M., JAKOB W., HUMPHREYS G.: Physically based rendering: From theory to implementation. Morgan Kaufmann, 2016. 7

[Poe21] POERSCHMANN J.: Mean estimation: Median of means, 2021. URL: https://github.com/jakobap/Median-of-Means-Estimator. 11

[SD12] SEN P., DARABI S.: On filtering the noise from the random parameters in monte carlo rendering. ACM Trans. Graph. 31, 3 (2012), 18–1. 3, 4

[SWZ96] SHIRLEY P., WANG C., ZIMMERMAN K.: Monte Carlo Techniques for Direct Lighting Calcultations. ACM Transactions on Graphics 15, 1 (1996), 1–36. 2

[VAN*19] VICINI D., ADLER D., NOVÁK J., ROUSSELLE F., BURLEY B.: Denoising deep monte carlo renderings. 316–327. URL: https://onlinelibrary.wiley.com/doi/abs/10.1111/cgf.13533, doi:10.1111/cgf.13533. 4

[Vea97] VEACH E.: Robust Monte Carlo methods for light transport simulation, vol. 1610. Stanford University PhD thesis, 1997. 2

[VG97] VEACH E., GUIBAS L. J.: Metropolis light transport. In Proceedings of the 24th annual conference on Computer graphics and interactive techniques (1997), pp. 65–76. 4

[VHH*19] VORBA J., HANIKA J., HERHOLZ S., MÜLLER T., KŘIVÁNEK J., KELLER A.: Path guiding in production. In ACM SIGGRAPH 2019 Courses. 2019, pp. 1–77. 4

[VRM*18] VOGELS T., ROUSSELLE F., MCWILLIAMS B., RÖTHLIN G., HARVILL A., ADLER D., MEYER M., NOVÁK J.: Denoising with kernel prediction and asymmetric loss functions. 1–15. URL: https://dl.acm.org/doi/10.1145/3197517.3201388, doi:10.1145/3197517.3201388. 4

[ZHD18] ZIRR T., HANIKA J., DACHSBACHER C.: Reweighting firefly samples for improved finite-sample monte carlo estimates. In Computer Graphics Forum (2018), vol. 37, Wiley Online Library, pp. 410–421. 3, 7

[Zhu20] ZHU S.: Survey: Machine Learning in Production Rendering. arXiv:2005.12518 [cs] (May 2020). arXiv: 2005.12518. 2