Reweighting Firefly Samples

February 26, 2019

I wrote this document as I read "Reweighting Firefly Samples for Improved Finite-Sample Monte Carlo Estimates" by Zirr, Hanika, and Dachsbacher [Zirr et al., 2018].

1 Basic Definitions

- Let F be an unbiased estimator for the rendering equation. That is:

$$E[F] = \int_\Omega f(X)\ \mathrm{d}X$$

  where Ω denotes the space of paths, X denotes a path, and $f(x)$ denote the path's measurement.

- We assume that, for each pixel, we trace N paths to compute the finite-sample estimate F^N given by:

$$F^N = \frac{1}{N}\sum_{i=1}^{n}\frac{f(X_i)}{p(X_i)}$$

  where X_i denotes the ith sampled path, and $p(X_i)$ denotes the probability of sampling X_i.

- We will also denote the individual contribution of each path X by

$$S(X) = \frac{f(X)}{p(X)}.$$

  In other words, we can write

$$F^N = \frac{1}{N}\sum_{i=1}^{N} S(X_i).$$

- Let $p^*(X)$ denote the ideal probability of sampling path X. In other words, $f(X)/p^*(X) = E[F]$.

- Let $r^*(X)$ be the ratio between the actual sample value $S(X) = f(X)/p(X)$ and the target value $E[F]$:

$$r^*(X) = \frac{S(X)}{E[F]} = \frac{f(X)/p(X)}{f(X)/p^*(X)} = \frac{p^*(X)}{p(X)}.$$

- The estimate then can be rewritten as:

$$F^N = \frac{1}{N}\sum_{i=1}^{N}\frac{f(X_i)}{p(X_i)} = \frac{1}{N}\sum_{i=1}^{N}\frac{f(X_i)}{p^*(X_i)/r^*(X_i)} = \frac{1}{N}\sum_{i=1}^{N}\frac{f(X_i)}{p^*(X_i)}r^*(X_i) = \frac{1}{N}\sum_{i=1}^{N}E[F]r^*(X_i).$$

- A firefly sample is a sample with high $r^*(X_i)$.

- Define a κ-firefly as a path sample with $r^*(X) \geq N/\kappa$. In particular, if we count the number of κ-fireflies and found more than κ of them, then the estimate F^N will be more than $E[F]$.
- In other word, we are very afraid of 1-fireflies. If there is one, our estimate overshoots $E[F]$.
- Let $\mathcal{F}_\kappa$ be the set of all κ-firefly:

$$\mathcal{F}_\kappa = \left\{ X \in \Omega \;\middle|\; r^*(X) \geq \frac{N}{\kappa} \right\} = \left\{ X \in \Omega \;\middle|\; S(X) \geq E[F]\frac{N}{\kappa} \right\}.$$

  Let $\mathcal{N}_\kappa$ be the set of paths that are not κ-fireflies: $\mathcal{N}_\kappa = \Omega - F_\kappa$.
- We can bound the probabiliy of sampling a κ-firefly as follows:

$$\Pr(X \in \mathcal{F}_\kappa) = \Pr\left(S(X) \geq E[F]\frac{N}{\kappa} \right) \leq \frac{\kappa}{N}.$$

  The last inequality is just Markov's inequality as $E[S] = E[F]$.

2 Reweighting Fireflies

- Consider the following estimator that weights the κ-fireflies differently:

$$S' = \begin{cases} S(X), & X \in \mathcal{N}_\kappa \\ w_\kappa(X)S(X), & X \in \mathcal{F}_\kappa \end{cases}$$

- Assume that $\Pr(X \in \mathcal{F}_\kappa) = \frac{\kappa}{N}$. Then, there is a constant $\alpha < 1$ where, if we set

$$w_\kappa(X) = \frac{\alpha N}{\kappa r^*(X)},$$

  then S' is an unbiased estimator of F. This is because:

$$\begin{aligned} E[S'] &= \Pr(X \notin \mathcal{F}_\kappa) E_{X \notin \mathcal{F}_\kappa}[S(X)] + \Pr(X \in \mathcal{F}_\kappa) E_{X \in \mathcal{F}_\kappa}[w(X)S(X)] \\ &= \frac{N-\kappa}{N} E_{X \notin \mathcal{F}_\kappa}[S(X)] + \frac{\kappa}{N} E_{X \in \mathcal{F}_\kappa}\left[\frac{\alpha N}{\kappa r^*(X)} S(X) \right] \\ &= E[F]\frac{N-\kappa}{N} E_{X \notin \mathcal{F}_\kappa}\left[\frac{p^*(X)}{p(X)} \right] + \alpha E[F] \\ &= E[F]\frac{N-\kappa}{N} \frac{\int_{\mathcal{N}_\kappa} p^*(X)\ \mathrm{d}X}{\Pr(X \in \mathcal{N}_\kappa)} \\ & E[F]\left(\frac{N-\kappa}{N} \frac{\int_{\mathcal{N}_\kappa} p^*(X)\ \mathrm{d}X}{\Pr(X \in \mathcal{N}_\kappa)} + \alpha \right). \end{aligned}$$

  That is, we can set

$$\alpha = 1 - \frac{N-\kappa}{N} \frac{\int_{\mathcal{N}_\kappa} p^*(X)\ \mathrm{d}X}{\Pr(X \in \mathcal{N}_\kappa)}$$

  to get $E[S'] = E[F]$. Now,

$$\alpha = 1 - \frac{N-\kappa}{N} \frac{\int_{\mathcal{N}_\kappa} p^*(X)\ \mathrm{d}X}{\Pr(X \in \mathcal{N}_\kappa)} \geq 1 - \int_{\mathcal{N}_\kappa} p^*(X)\ \mathrm{d}X.$$

  So, $\alpha \leq 1$.

- While it is possible (under an assumption) to weight the firefly down to still get an unbiased estimatior, the paper says it would use $\alpha = 1$ to keep as much energy as possible. This makes sense because there will be no way we would sample exactly κ fireflies.
- What we now need is (1) how to classify a sample as a κ-firefly, and (2) how to estimate $r^*(X)$ so that we can weight the sample down by $N/(\kappa r^*(X))$.
- One thing that is not clear in the paper is that the weight function is not spelled out as an equation. I think it is:

$$w(S_i) = \min\left\{1, \frac{N}{\kappa r^*(X_i)}\right\}.$$

  In this way, we make sure that we only decrease the estimate that we compute.

3 Firefly Classification by Sampling Counting

- Suppose we have sampled N samples $S_1, S_2, \ldots, S_N$. For each sample S_i, the paper suggests computing n_i, the number of samples that are similar to S_i as follows:

$$n_i = \sum_{j=1}^{N} K(S_i - S_j)$$

  where K is a kernel function with $K(0) = 1$ that falls off to zero on both sides.
- For the ith sample, suppose that we find $n_i \geq \kappa$ samples similar to S_i. Then, there is a good chance that $X_i \in \mathcal{N}_{n_i}$.

  To see this, we shall bound the probability that $X_i \in \mathcal{F}_{n_i/\beta}$ for $\beta \geq 1$ given that n_i samples similar to X_i are generated. Let I_j be the probability that the jth sample belongs to $\mathcal{F}_{n_i/\beta}$. We have that $\Pr(I_j = 1) \leq n_i/(\beta N)$ by Markov's inequality. Because the ith sample belongs to $\mathcal{F}_{n_i/\beta}$, it means that all the n_i samples similar to it must belong to $\mathcal{F}_{n_i/\beta}$ too. This probability is thus bounded above by the probability that $\Pr(\sum I_j \geq n_i)$. We have that:

$$\Pr(\sum I_j \geq n_i) = \Pr\left(\sum I_j \geq \frac{n_i}{N} N\right) = \Pr\left(\sum I_j \geq \beta \frac{n_i}{\beta N} N\right).$$

  Using Theorem 1 in [Arratia and Gordon, 1989] with $p = n_i/(\beta N)$ and $a = \beta p = \beta n_i/(\beta N)$, we have that:

$$\Pr\left(\sum I_j \geq n_i\right) \leq e^{-NH}$$

  where

$$H = a \log \frac{a}{p} + (1-a) \log \frac{1-a}{1-p}.$$

  Now,

$$\begin{aligned} NH &= Na \log \frac{a}{p} + N(1-a) \log \frac{1-a}{1-p} \\ &= n_i \log \beta + (N - n_i) \log \frac{N(1-a)}{N(1-p)} \\ &= n_i \log \beta + (N - n_i) \log \frac{N - n_i}{N - n_i/\beta}. \end{aligned}$$

So,

$$\begin{aligned}
e^{-NH} &= \exp\Big(- n_i \log\beta - (N-n_i)\log\frac{N-n_i}{N-n_i/\beta}\Big) \\
&= \exp(-n_i\log\beta)\left(\frac{N-n_i/\beta}{N-n_i}\right)^{N-n_i} \\
&= \exp(-n_i\log\beta)\left(1+\frac{n_i-n_i/\beta}{N-n_i}\right)^{N-n_i} \\
&\le \exp(-n_i\log\beta)\exp\left(n_i-\frac{n_i}{\beta}\right) \\
&= \exp\Big(- n_i\log\beta + n_i - n_i/\beta\Big) \\
&= \exp\Big(- n_i\Big(\log\beta+\frac{1}{\beta}-1\Big)\Big).
\end{aligned}$$

The value $\log\beta + \frac{1}{\beta} - 1$ is greater than 0 for all $\beta > 1$.

As a result, as n_i increases, it becomes exponentially less likely for X_i to belong to any $\mathcal{F}_{n'}$ where $n' > n_i$. (Just graph it.)

While it is not said in the paper, I think we may also bound the probability that $X_i \in \mathcal{N}_{n'}$ where $n' < n_i$ and obtain an exponential decay as well.

- Now, to reweight the ith sample, we estimate $r^*(S_i)$ by N/n_i. Why? Because N/n_i is the most likely value according to the last item.

- The paper says that we may expect one sample in $\mathcal{F}_1$. When we get this sample, we will have that $n_i = 1$. Since there is only one sample, we don't know the right $\kappa \le 1$ such that X_i belongs to $\mathcal{F}_\kappa$. As a result, we cannot sensibly reweight this sample. Because of this, the paper recommends discarding this sample.

- In general, instead of discarding just only one sample, the paper sets up a parameter $\kappa_{\min}$ such that it discards X_i if $n_i \le \kappa_{\min}$. Typically, $\kappa_{\min} = 1$. To minimize probability of misclassification, the paper averages n_i of the immediate neighboring pixels (this doesn't make sense now, but it will later) and use this to decide whether to discard the sample.

- By the introduction of $\kappa_{\min}$, the ratio takes the form:

$$r_c^*(X_i) = \frac{N}{n_i - \kappa_{\min}},$$

and the weighting function takes the form:

$$w_c(X_i) = \min\left\{1, \frac{N}{\kappa}\frac{n_i-\kappa_{\min}}{N}\right\} = \min\left\{1, \frac{n_i-\kappa_{\min}}{\kappa}\right\}$$

- The weight w_c, however, treats samples with low values but small counts the same way as those with high values and small counts. That is, if the counts are small, the samples will be downweighted. This can lead to considerable energy loss.

- To fix the above problem, they would like to compute a lower bound of the expected value $E_{\min}[F]$. This would then be used to compute an upper bound $r_v^*(S_i) = S_i/E_{\min}[F]$ to clamp $r_c^*(S_i)$. In other words, the ultimate weighting function is given by:

$$w(S_i) = \min\left\{1, \frac{N}{\kappa\min\{r_c^*(X_i), r_v^*(X_i)\}}\right\} = \min\left\{1, \max\left\{\frac{n_i-\kappa_{\min}}{\kappa}, \frac{NE_{\min}[F]}{\kappa S_i}\right\}\right\}.$$

- When accumulating the samples, the algorithm sorts the samples based on their values in increasing order. That is, say, $S_1 \leq S_2 \leq \cdots \leq S_N$. Then, it computes the partial $F_i'^N$ that takes into account samples up to S_i. When computing $F_i'^N$, it uses $F_{i-1}'^N$ as $E_{\min}[F]$. That is:

$$\begin{aligned} F_1'^N &= \frac{1}{N} w_c(S_1) S_1 \\ &= \frac{1}{N} \min\left\{1, \frac{n_1 - \kappa_{\min}}{\kappa}\right\} S_1 \\ F_i'^N &= F_{i-1}'^N + \frac{1}{N} w(S_i) S_i \\ &= F_{i-1}'^N + \frac{1}{N} \min\left\{1, \max\left\{\frac{n_i - \kappa_{\min}}{\kappa}, \frac{N F_{i-1}'^N}{\kappa S_i}\right\}\right\} S_i. \end{aligned}$$

- I don't understand why they don't just sort the values and reweight only the high value ones instead of doing some complicated stuff here. The last section sounds like a hack more than principled reasoning.

4 Practical Implementation

- The algorithm outlined in the last section requires us to keep all the samples for a pixel. This requires a lot of memory. The paper comes up with a scheme that does not have to store all individual samples.
- The algorithm employs a fixed number of framebuffers, each of such a buffer is called a *cascade*. The buffer B_j accumulates samples with brighness in the range $[b^{j-1}, b^{j+1}]$, where b is a small constant. The paper typically uses $b = 8$.
- The total number of required buffers depends on the maximum sample brightness:

$$j \leq \log_b S_{\max}.$$

  Typically, $S_{\max}$ should be the brighness of the brightness light source.
- The paper seems to say that B_0 is the first buffer. However, we can go lower to B_{-1}, B_{-2}, and so on. I'm not so sure how low should be we down.
- Given a sample, we first compute

$$j = \lfloor \log_b S_i \rfloor.$$

  We then splat S_i into B_j and B_{j+1}. When splatting to B_j, the sample is multiplied with the weight:

$$\alpha_j = \frac{b^j/S_i - 1/b}{1 - 1/b}.$$

  When splatting to B_{j+1}, the weight is $\alpha_{j+1} = 1 - \alpha_j$.
- For samples that is dimmer than the range of the lowest buffer, we splat it fully to the lowest buffer.
- With the weights, we can have an estimate of the number of samples in each buffer. However, the paper does not do this. I don't know why.
- The number of samples stored in B_j can be estimated by:

$$n_j = \frac{N B_{j-1}}{b^{j-1}} + \frac{N B_j}{b^j} + \frac{N B_{j+1}}{b^{j+1}}.$$

- When computing the estimate for each pixel, the algorithm iterates from the lowest buffer to the highest buffer. Using n_j to compute the weighting factor. However, instead of computing $r^* v(S_i)$ as $B_j / E_{\min} F$, it computes $r^* v(S_i)/b^j$ instead, and I don't know why either.

Like, the promise of this paper is exciting, but many parts seem unclear and extremely hacking. Decisions are unmotivated, and logic does not flow well at all.

The GLSL source code in the website of the project also contains code that does not conform to what is written in the paper. What the heck!!!

References

[Arratia and Gordon, 1989] Arratia, R. and Gordon, L. (1989). Tutorial on large deviations for the binomial distribution. pages 125–131.

[Zirr et al., 2018] Zirr, T., Hanika, J., and Dachsbacher, C. (2018). Re-weighting firefly samples for improved finite-sample monte carlo estimates. *Comput. Graph. Forum*, 37(6):410–421.