A MIS and AMIS

A.1 Multiple Importance Sampling

Multiple importance sampling (MIS) is a strategy to combine several sampling models using a weighting function. The weighting function $w_t(\boldsymbol{\omega})$ of the tth sampling model using balance heuristic is given as

$$w_t(\boldsymbol{\omega}) = \frac{N_t p_t(\boldsymbol{\omega})}{\sum_{k=0}^{T-1} N_k p_k(\boldsymbol{\omega})}, \tag{A.1}$$

where $\boldsymbol{\omega}$ is a sample point (in our case, ray direction), N_t the number of samples, p_t the PDF, and T the number of sampling models.

A.2 Adaptive Multiple Importance Sampling

AMIS is aimed at optimally recycling past simulations in an iterative importance sampling scheme. The difference to earlier adaptive importance sampling methods is that the past weighting functions are recomputed by MIS at each iteration. After t iterations, the weighting function $w_j(\boldsymbol{\omega})$ ($0 \leq j \leq t$) is given as

$$w_j(\boldsymbol{\omega}) = \frac{N_j p(\boldsymbol{\omega}; \hat{\boldsymbol{\theta}}_j)}{\sum_{k=0}^{t} N_k p(\boldsymbol{\omega}; \hat{\boldsymbol{\theta}}_k)}, \tag{A.2}$$

where the PDF p at the jth iteration is parameterized by $\hat{\boldsymbol{\theta}}_j$. The next parameter $\hat{\boldsymbol{\theta}}_{t+1}$ is estimated from past samples. It will be described in next section. Let $f(\boldsymbol{\omega})$ be the integrand, a weighted value of ith sample at jth iteration is obtained as

$$s_{i,j} = w_j(\boldsymbol{\omega}_{i,j}) \frac{f(\boldsymbol{\omega}_{i,j})}{N_j p(\boldsymbol{\omega}_{i,j}; \hat{\boldsymbol{\theta}}_j)}. \tag{A.3}$$

And, the integral is estimated as

$$\int_{\Omega} f(\boldsymbol{\omega}) d\boldsymbol{\omega} \approx \sum_{j=0}^{t} \sum_{i=0}^{N_j-1} s_{i,j}. \tag{A.4}$$

The performance of AMIS depends on an updating scheme. In this paper, we propose an appropriate method for final gathering.

B Estimation of Optimal Parameter

We estimate the optimal $\hat{\boldsymbol{\theta}}_{t+1}$ from past t samples with MLE. In this paper, we use following likelihood function:

$$L_t(\boldsymbol{\theta}) = \prod_{j=0}^{t} \prod_{i=0}^{N_j-1} p(\boldsymbol{\omega_{i,j}}; \boldsymbol{\theta})^{s_{i,j}}. \tag{B.1}$$

By maximizing $L_t(\boldsymbol{\theta})$, we obtain $\hat{\boldsymbol{\theta}}_{t+1}$. For convenience, we use the log-likelihood function given as

$$\begin{aligned} l_t(\boldsymbol{\theta}) &= \sum_{j=0}^{t} \sum_{i=0}^{N_j-1} s_{i,j} \log p(\boldsymbol{\omega_{i,j}}; \boldsymbol{\theta}) \\ &= \sum_{j=0}^{t} \sum_{i=0}^{N_j-1} s_{i,j} \log(\frac{\alpha+1}{2} |\boldsymbol{\omega_{i,j}} \cdot \boldsymbol{n}|^{\alpha}) \\ &= \sum_{j=0}^{t} \sum_{i=0}^{N_j-1} s_{i,j} \log(\frac{\alpha+1}{2}) + \alpha s_{i,j} \log |\boldsymbol{\omega_{i,j}} \cdot \boldsymbol{n}|. \end{aligned} \tag{B.2}$$

Solving

$$0 = \frac{\partial l_t(\boldsymbol{\theta})}{\partial \alpha} = \sum_{j=0}^{t} \sum_{i=0}^{N_j-1} \frac{s_{i,j}}{\alpha+1} + s_{i,j} \log |\boldsymbol{\omega_{i,j}} \cdot \boldsymbol{n}|, \tag{B.3}$$

results in

$$\hat{\alpha}_{t+1} = -1 - \frac{\sum_{j=0}^{t} \sum_{i=0}^{N_j-1} s_{i,j}}{\sum_{j=0}^{t} \sum_{i=0}^{N_j-1} s_{i,j} \log |\boldsymbol{\omega_{i,j}} \cdot \boldsymbol{\hat{n}_{t+1}}|}. \tag{B.4}$$

Similary, solving

$$0 = \frac{\partial l_t(\boldsymbol{\theta})}{\partial \boldsymbol{n}} = -\alpha \sum_{j=0}^{t} \sum_{i=0}^{N_j-1} s_{i,j} \tan(\arccos(\boldsymbol{\omega_{i,j}} \cdot \boldsymbol{n})), \tag{B.5}$$

yields $\hat{\boldsymbol{n}}_{t+1}$. However, this is computationally expensive. Therefore, we assume that $\hat{n}_{t+1}$ is the average of the sampled directions, which is given as

$$\hat{\boldsymbol{n}}_{t+1} = \frac{\sum_{j=0}^{t} \sum_{i=0}^{N_j-1} s_{i,j} \boldsymbol{\omega}_{i,j}}{\| \sum_{j=0}^{t} \sum_{i=0}^{N_j-1} s_{i,j} \boldsymbol{\omega}_{i,j} \|}. \tag{B.6}$$

C Pseudo Code

Algorithm 1 is the pseudo code of our method. The procedure Sample() returns a random sampled direction according to the PDF $p(\omega; \hat{\boldsymbol{n}}_t, \hat{\alpha}_t)$. The procedure Trace() evaluates the integrand by tracing a final gather ray. The procedure NextNormal() and NextAlpha() are the same as Equation (B.6) and Equation (B.4) respectively. The denominators of Equation (A.2) are accumlated into the variable d.

The computation order is $O(TM)$, where M is the total number of samples. To reduce the computation time, we are bound to use a small number of iterations. It is more efficient to use only recent samples instead of all samples generated in the past.

Algorithm 1 Final Gathering using AMIS

```
procedure FinalGather(n_s, T, N)
  n̂_0 = n_s
  α̂_0 = 1
  for t = 0 to T − 1 do
    for i = 0 to N_t − 1 do
      ω_{i,t} = Sample(n̂_t, α̂_t)
      f_{i,t} = Trace(ω_{i,t})
    end for
    Weight(s, d, n̂, α̂, ω, f, t, N)
    n̂_{t+1} = NextNormal(s, ω, t, N)
    α̂_{t+1} = NextAlpha(s, ω, n̂_{t+1}, t, N)
  end for
  return Sum(s, T − 1, N)

procedure Weight(s, d, n̂, α̂, ω, f, t, N)
  for j = 0 to t − 1 do
    for i = 0 to N_j − 1 do
      d_{i,j} += N_t p(ω_{i,j}; n̂_t, α̂_t)
    end for
  end for
  for i = 0 to N_t − 1 do
    d_{i,t} = 0
    for k = 0 to t do
      d_{i,t} += N_k p(ω_{i,t}; n̂_k, α̂_k)
    end for
  end for
  for j = 0 to t do
    for i = 0 to N_j − 1 do
      s_{i,j} = f_{i,j} / d_{i,j}
    end for
  end for

procedure Sum(s, t, N)
  S = 0
  for j = 0 to t do
    for i = 0 to N_j − 1 do
      S += s_{i,j}
    end for
  end for
  return S
```