多光源レンダリング法のための効率的な可視関数の確率的評価

Efficient Probabilistic Visibility Evaluation for Many-light Rendering

名畑豪祐* 岩崎慶* 土橋宜典† 西田友是‡

Kosuke NABATA* Kei IWASAKI* Yoshinori DOBASHI† and Tomoyuki NISHITA‡

*和歌山大学 *Wakayama University

†北海道大学/JST CREST †Hokkaido University /JST CREST

‡広島修道大学/UEI リサーチ ‡Hiroshima Shudo University/UEI Research

1 はじめに

CG 分野において，効率的な可視判定法の研究は多くされている．特に，シーンをレンダリングする際に，各ピクセルに対応するシーン上の陰影を計算する点であるシェーディング点が光源に照らされているかどうかを判定するために，光源とシェーディング点間の可視判定が大量に必要となり，可視判定が大きなボトルネックとなる．可視判定を行うには，2 点間を結ぶ直線とシーン中の物体との交差判定を行う必要があり，最先端の高速化データ構造や GPU を使用しても非常に計算コストが高い処理である．

1 点対多点の可視判定を効率よく行う方法としてシャドウマッピング法がある．光源を視点としてシーンをラスタライズし，光源までの距離を格納した深度マップを生成する．可視判定はシェーディング点と光源間の距離と，シェーディング点に対応する深度マップのピクセル値との比較で行える．しかしながら，画像というデータ構造を使用しているため，エイリアシングの問題がある．

近年，多数の仮想的な点光源を設置し，点光源からの寄与を計算することによって大域照明効果を効率的に計算する多光源レンダリング法がよく研究されている．この手法ではレンダリング時間の内，可視判定のための時間が支配的である．そこで，本研究では，多光源レンダリング法において確率的に可視判定を行うことで，可視判定を行う回数を削減し効率的に画像を生成する手法を提案する．提案法では，ある確率で可視判定を省略し，それ以外では実際に可視判定を行う．そのため，可視判定を省略する確率を大きくするほど，可視判定を行う回数を削減でき，高速に画像を生成できる．しかし，確率を大きくするほど確率的評価による分散が増加し画像にノイズが発生する．そこで，可視判定を省略する確率と，確率的評価による分散の関係を考慮し，画像生成の効率を最大とする確率の導出を行う．また，本研究では可視関数の空間，方向のコヒーレンスを利用するために，可視判定結果のキャッシングを行う．提案法による確率的な可視関数評価により，可視関数の値を正確に評価した場合と比較して最大 6 倍高速化された．また，従来の確率的な可視関数評価法と比較して最大 2.5 倍高速化された．

2 研究背景

2.1 先行研究

Keller らは，少量の仮想的な点光源(Virtual Point Light, VPL)を使用し間接照明を計算するインスタントラジオシティ法を提案した[1]．多光源レンダリング法は，インスタントラジオシティ法を基にした手法で，大量の VPL を使用してレンダリングする手法である．多光源レンダリング法は，大域照明効果をよく近似でき，効率的に写実的な画像を生成できる手法として研究されてきている[2]．Walter らは，VPL の階層構造を使用することで効率的なレンダリングを行う方法を提案した[3]．Hasan らは，多光源問題を行列で表し，行，列サンプリングにより効率的にレンダリングを行う方法を提案した[4]．Ou らはこの手法を拡張し，シェーディング点をクラスタリングし，各クラスタで行・列サンプリングを行うことでより効率的にレンダリングを行う方法を提案した[5]．
Georgiev らは，VPL からの寄与を記録するキャッシュ点を使うことで，VPL の寄与に比例した重点的サンプリングを行う方法を提案した[6]．これらの手法は，すべての VPL からの寄与を計算するのではなく，寄与の高い VPL を重点的に使用することで効率的にレンダリングを行っている．しかし，依然として大量の可視判定が必要であり，シャドウマッピング法またはレイキャスティング法を用いて可視判定を行っている．提案法では，これらの手法とは別のアプローチにより効率化を行うため，これらの手法と組み合わせることができる．

Clarberg らは，直接照明からの寄与計算において，可視判定結果を記録するキャッシュ点を用いることによって，効率的にレンダリングを行う手法を提案した[7]．この手法では，制御変量法を用いることで，ノイズの少ない画像を生成する．Popov らは，類似する経路間で可視判定結果を再利用することで，可視判定回数を削減する手法を提案した[8]．Ulbrich らは，低解像度のシャドウマップをキャッシュ点として使用し，キャッシュ点の重要度に基づき，キャッシュ点の位置を改善していくことで，プログレッシブなレンダリングを行う手法を提案した[9]．

Billen らは，可視関数を確率的に評価し，交差判定を行う回数を削減する方法を提案した[10]．この手法では直接照明のみを扱っており，間接照明は考慮していなかった．また，画像生成の効率についても考慮していなかった．Veach は，画像生成の効率を考慮したロシアンルーレット法により，可視判定を行う回数を削減する方法を提案した[11]．この方法は，双方向パストレーシングにおいて導入された手法であるため，保守的な仮定を多く立てており，高い高速化率は得られない．

本稿では，Veach の提案した，画像生成を考慮したロシアンルーレット法を，多光源レンダリング法に応用し，キャッシュ点を用いることでさらなる効率化を行う方法を提案する．Veach の手法では，寄与の低い VPL に対して可視判定を省略する確率が高くなるのに対し，提案法では，寄与の低い VPL だけでなく，近傍で可視判定の結果が類似している VPL に対しても可視判定を省略する確率を高く設定することができる．これにより，Veach の手法では，ある程度の回数の可視判定が必ず必要であるのに対し，提案法では，可視判定の回数を著しく削減することが可能となる．

2.2　多光源レンダリング法 による輝度計算

シェーディング点xから視点x_v方向に出射する放射輝度L_oは以下の式で計算される．

$$L_o(x,x_v)=\int_A L(y,x)f_r(y,x,x_v)V(x,y)G(x,y)dA(y) \qquad (1)$$

ここで，Aはシーン内の物体表面全体の集合，$L(y,x)$はシーン中の物体表面上の位置yからシェーディング点xに向かう放射輝度，f_rは BRDF，$V(x,y)$は可視関数，$G(x,y)$は幾何項である．可視関数$V(x,y)$は点xと点yが互いに可視なら1，可視でないなら0を返す関数であり，可視判定を行うことで評価できる．多光源レンダリング法では，シーン内の物体表面全体からの寄与の積分を，大量の VPL からの寄与Lf_rVGの総和で近似する．

$$L_o(x,x_v)=\sum_{i=1}^{N} L(y_i,x)f_r(y_i,x,x_v)V(x,y_i)G(x,y_i) \qquad (2)$$

ここで，y_iはi番目の VPL，Nは VPL の数を表す．本研究では，$L(y_i,x)f_r(y_i,x,x_v)G(x,y_i)$を$i$番目の VPL からの暫定的な寄与$t_i$と定義する．また，表記簡略化のため$V(x,y_i)$を$v_i$と表記する．

3　提案法

提案法の処理の流れを図 1 に示す．提案法は前処理，レンダリング処理から成る．前処理では，まず VPL を生成し，その後キャッシュ点を生成する．VPL は従来と同様の方法[2]で生成し，キャッシュ点にはシェーディング点の中から選択したものを使用する．キャッシュ点では各 VPL に対する可視関数の値と，各 VPL からの寄与の分散を記録する．レンダリング処理では，まずシェーディング点の近傍のキャッシュ点を計算する．そして，各 VPL に対して，キャッシュ点の情報から推定した可視関数の値と，VPL から

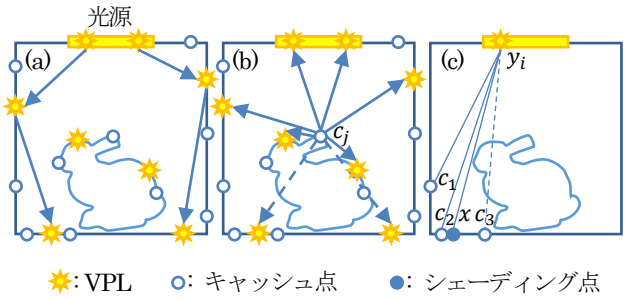

図 1: 提案法の処理の流れ．(a)VPL，キャッシュ点の生成．(b)各キャッシュ点c_jで各 VPLy_iに対して可視関数の値とt_iv_iの分散を記録．(c)近傍のキャッシュ点(c_1,c_2,c_3)の情報から推定した可視関数の値$\hat{V}(x,y_i)$を使い放射輝度L_oを計算．

の暫定的な寄与t_iとの積を計算し，視点方向へ出射する放射輝度L_oを計算する．

3.1　キャッシュ点の生成

本研究では Ou らの手法と同様，6 次元の kd 木を使用したシェーディング点のクラスタリングによりキャッシュ点を生成する[5]．シェーディング点を位置(3 次元)，法線(3 次元)の 6 次元点として表し，全てのシェーディング点を包含する軸平行の 6 次元バウンディングボックスを計算する．そして，含まれるシェーディング点の数が十分小さくなるか，バウンディングボックスが十分小さくなるまで，バウンディングボックスの辺の長さが最大の軸の中点で分割を行う．分割された各クラスタからランダムに選択したシェーディング点をキャッシュ点として使用する．

各キャッシュ点では，すべての VPL に対して可視判定を行い可視関数の値を記録する．また，可視判定を省略する確率の計算のために各 VPL からの寄与の分散を計算し記録しておく．

3.2　可視関数の確率的評価

提案法では，シェーディング点xと VPL y_i 間の可視関数を以下の式で推定する．

$$\hat{V}(x,y_i)=\begin{cases}\dfrac{\alpha}{p_i} & \text{確率}p_i\text{で選択}\\ \dfrac{v_i-\alpha}{1-p_i} & \text{それ以外}\end{cases} \qquad (3)$$

ここで，αは任意のパラメータである．提案法では確率p_iで可視判定を省略し，α/p_iを可視関数の推定値とする．そして，確率$1-p_i$で可視判定を行い，$(v_i-\alpha)/(1-p_i)$を可視関数の推定値とする．VPLy_iからの寄与の期待値$E\left[t_i\hat{V}(x,y_i)\right]$，分散$Var\left[t_i\hat{V}(x,y_i)\right]$は以下の式で計算される．

$$E\left[t_i\hat{V}(x,y_i)\right]=t_i\left(p_i\frac{\alpha}{p_i}+(1-p_i)\frac{v_i-\alpha}{1-p_i}\right)=t_iv_i \qquad (4)$$

$$Var\left[t_i\hat{V}(x,y_i)\right]=t_i^2\frac{(p_iv_i-\alpha)^2}{p_i(1-p_i)} \qquad (5)$$

式(4)より，推定値の期待値と真値が等しいので$t_i\hat{V}(x,y_i)$はt_iv_iの不偏推定量である．式(5)より，α=p_iv_iのとき確率的評価による分散は0となる．すなわち，$\alpha = p_iv_i$とできるならば，可視判定を省略する確率p_iを1に近づけても分散は0である．しかし，v_iの値は実際に評価するまで未知である．そこで，近傍のキャッシュ点から可視関数の値を補間した値q_iを用い，$\alpha = p_iq_i$とする．キャッシュ点からの補間については，Clarberg らの手法を使用する[7]．式(3)，式(5)に$\alpha = p_iq_i$を代入した式は以下となる．

$$\hat{V}(x,y_i) = \begin{cases} q_i & \text{確率}p_i\text{で選択} \\ \dfrac{v_i - p_iq_i}{1-p_i} & \text{それ以外} \end{cases} \tag{6}$$

$$Var\left[t_i\hat{V}(x,y_i)\right] = t_i^2\frac{p_i(v_i-q_i)^2}{1-p_i} \tag{7}$$

3.3　確率p_iの導出

提案法では，可視判定を省略する確率p_iを大きくするほど高速に画像を生成できるが，確率的評価による分散が増加し，画像にノイズが生じる．一方，p_iを小さくすれば，分散は小さくできるが，多くの可視判定が行われ計算時間が増加する．そこで，Veach の手法と同様に画像生成の効率を最大とするためにσ^2nを最小とする確率p_iを使用する[11]．ここで，σ^2はサンプルの分散，nはサンプル当たりのレイの平均追跡回数である．提案法では１つのサンプルは１つの VPL に対応する．実際に全 VPL について可視判定を行った場合の VPL からの寄与の分散をσ_0^2，サンプル当たりのレイの平均追跡回数をn_0とすると，提案法におけるサンプルの分散σ^2は，実際に全 VPL について可視判定を行った場合の VPL からの寄与の分散σ_0^2と確率的評価による分散の増加量の和であり，以下の式で計算される．

$$\sigma^2 = \sigma_0^2 + t_i^2\frac{p_i(v_i-q_i)^2}{1-p_i} \tag{8}$$

ここで，σ_0^2はキャッシュ点から補間した値を用いる．サンプル当たりのレイの平均追跡回数nについては，提案法では確率p_iで可視判定を省略するので，サンプル当たりのレイの追跡回数は平均でp_i減少する．よって，σ^2nは以下の式で計算される．

$$\left(\sigma_0^2 + t_i^2\frac{p_i(v_i-q_i)^2}{1-p_i}\right)(n_0 - p_i) \tag{9}$$

式(9)をp_iについて微分し，結果を0とおくと，σ^2nを最小とする確率p_iは以下の式で計算される．

$$p_i = 1 - \sqrt{\frac{A_i(n_0-1)}{\sigma_0^2 - A_i}} \tag{10}$$

ここで，$A_i = t_i^2(v_i-q_i)^2$である．パラメータαの計算時と同様，v_iは実際に評価するまで未知であるためA_iを計算することができない．そこで，$(v_i-q_i)^2$を近傍のキャッシュ点c_jの可視関数の値の分散$Var\left[V(c_j,y_i)\right]$で置き換え，$A_i = t_i^2Var\left[V(c_j,y_i)\right]$とする．可視関数の補間値$q_i$が0か1に近いとき，補間値$q_i$は可視関数$v_i$を精度よく近似していると考えられるため，このとき$(v_i-q_i)^2$は小さくなるべきである．キャッシュ点での可視関数の値の分散はこの性質を満たすため分散を近似として用いた．

4　実装の詳細

4.1　パラメータの計算

可視関数の補間値q_iが0または1と計算された場合，キャッシュ点の可視関数の分散$Var\left[V(c_j,y_i)\right]$は0となる．この場合，$A_i = t_i^2Var\left[V(c_j,y_i)\right]$は0と計算される．そして，式(10)より，可視判定を省略する確率p_iは1と計算される．もし可視関数の補間値q_iが実際の可視関数の値v_iと異なる場合，可視判定が行われないため$t_i\hat{V}(x,y_i)$の期待値はt_iv_iに収束しない．そのため，$Var\left[V(c_j,y_i)\right]$の下限をクランプした以下の式で$A_i$を計算する．

$$A_i = t_i^2\max\left(\epsilon_1, Var\left[V(c_j,y_i)\right]\right) \tag{11}$$

ここで．ϵ_1は 0 より大きい値で，0.1 程度にすればよい結果が得られている．A_iがσ_0^2より大きいとき，$\sqrt{A_i(n_0-1)/(\sigma_0^2-A_i)}$ が負の数の平方根となるが，この場合は可視判定を省略する確率p_iを0に設定すればよい．また，可視判定を省略する確率p_iが負の値をとらないよう下限をクランプし，可視判定を省略する確率p_iが大きすぎる場合は，補間値q_iと実際の可視関数の値v_iが大きく異なる場合に大きな分散が生じるのを防ぐために上限をクランプする．よって，可視関数を省略する確率p_iは以下の式で計算する．

$$p_i = \min(1-\epsilon_2, \max(0, p_i)) \tag{12}$$

ここで，ϵ_2は 0 より大きく 1 以下の値で，0.1 程度にすればよい結果が得られている．

Veach の論文で述べられているように，サンプル当たりのレイの追跡回数n_0はすべての種類のレイを含める．N個の VPL からの寄与を計算する場合のレイの追跡回数は，視点からのレイが拡散面と交差するまでのレイの追跡回数n_dと，シェーディング点と各 VPL 間の可視判定のためのレイの追跡回数Nの和となる．よって，サンプル当たりのレイの追跡回数n_0は以下の式で計算される．

$$n_0 = \frac{n_d}{N} + 1 \tag{13}$$

4.2　負値の可視関数

提案法による可視関数の推定値は0か1ではなく，負の値も取りうるため，ピクセル値が負となる場合がある．現在の実装では，ピクセル値が負となった場合は，ピクセル値を0でクランプしている．そのため，真値より明るい画像が生成されていることになる．しかしながら，実験では，値が負となるピクセルはわずかであり，また，値も非常に0に近く，クランプによるアーティファクトも見られない．

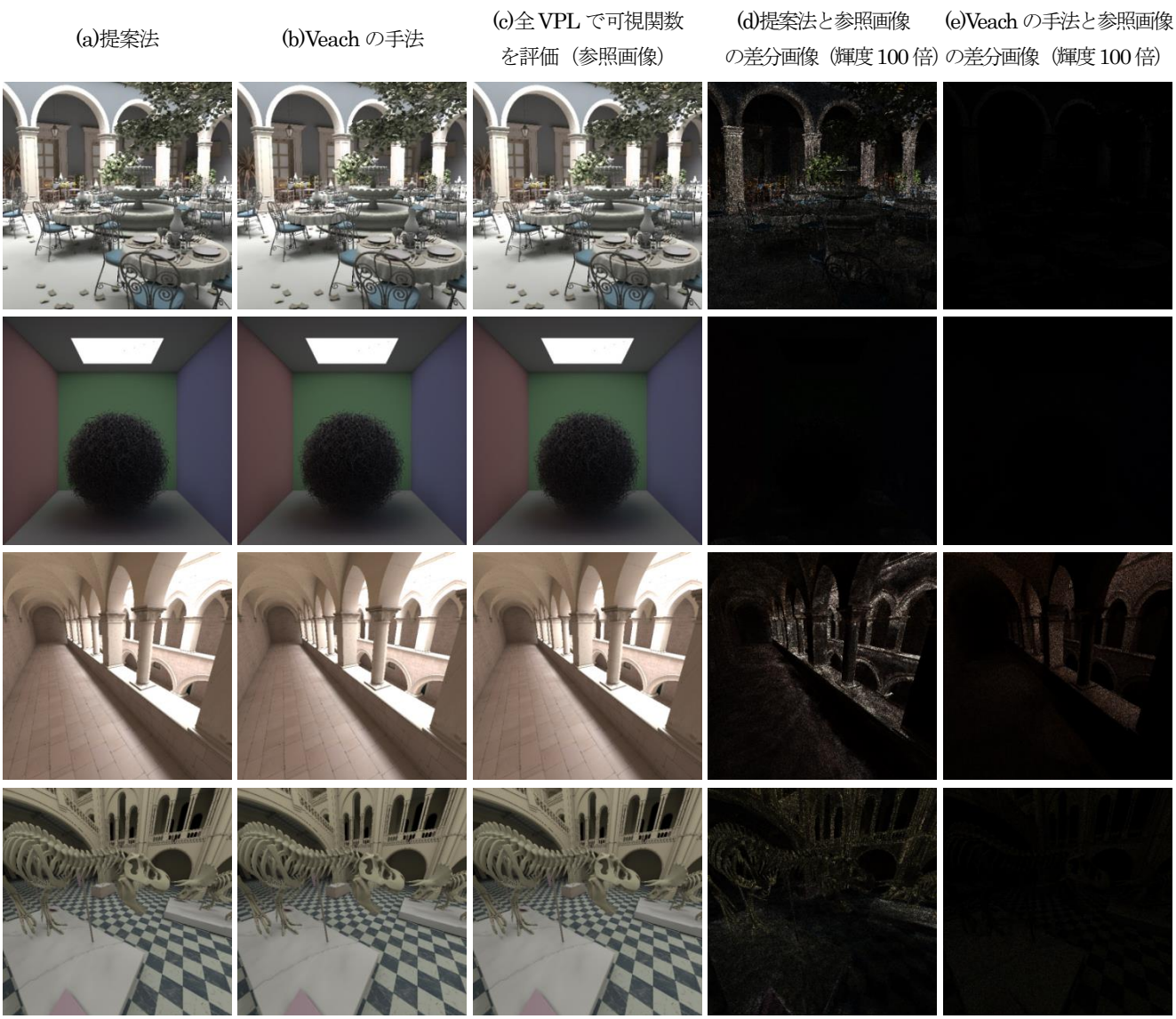

図 2: Sanmiguel, Hairball, Sponza, Museum シーンの同 VPL 数でのレンダリング結果.

表 1: 図 2 レンダリング時の計算時間と RMSE.
括弧内の数字は全 VPL で可視関数を評価した場合(c)に対する計算時間の高速化率を表す.

シーン	三角形数	VPL 数	キャッシュ点数	計算時間（秒）			RMSE	
				(a)提案法	(b)Veach の手法	(c)	(a)提案法	(b)Veach の手法
Sanmiguel	1.8M	550k	1,885	2,417 (×2.60)	5,204 (×1.21)	6,273	$1.70e^{-3}$	$1.09e^{-4}$
Hairball	2.8M	100k	500	933 (×6.08)	2,310 (×2.45)	5,668	$1.43e^{-4}$	$5.72e^{-5}$
Sponza	65K	300k	1,414	916 (×2.12)	1,320 (×1.47)	1,942	$2.61e^{-3}$	$1.19e^{-3}$
Museum	1.4M	100k	1,900	488 (×2.56)	858 (×1.45)	1,247	$8.81e^{-4}$	$2.75e^{-4}$

5 実験結果

実験には，Xeon E5-2680v2 2.8GHz CPU，64GB メモリを搭載した PC を使用し，マルチスレッドによる並列化を行っている．レンダリング画像の解像度は全て 512×512 である．使用した近傍のキャッシュ点の数は 3 である．近傍のキャッシュ点数を変えて実験を行ったが計算時間，誤差の指標である二乗平均平方根誤差 (Root Mean Square Error, RMSE)への影響は小さく，3 程度で十分である．なお，VPL を使用したレンダリングでは，幾何項の特異点によるアーティファクトが発生するが, 実験ではクランプにより緩和している．

表 2: 図 2 レンダリング時のピクセル当たりの
平均可視判定回数（削減率）

シーン	平均可視判定回数（削減率）	
	(a)提案法	(b)Veach の手法
Sanmiguel	65,185 (88%)	196,818 (64%)
Hairball	10,401 (90%)	37,371 (63%)
Sponza	31,485 (90%)	78,820 (74%)
Museum	9,724 (90%)	16,517 (83%)

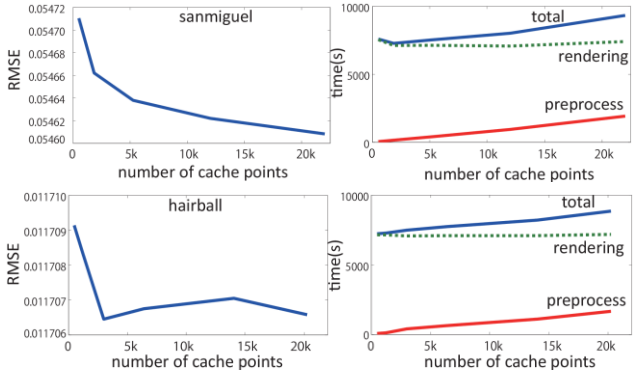

図 3: キャッシュ点の数と，パストレーシングで生成した画像との RMSE の関係を表すグラフ（左図）と，キャッシュ点の数と計算時間の関係を表すグラフ（右図）．上段は Sanmiguel シーン，下段は Hairball シーン．Intel Xeon W5590 3.33GHz CPU を搭載した PC を使用し計測.

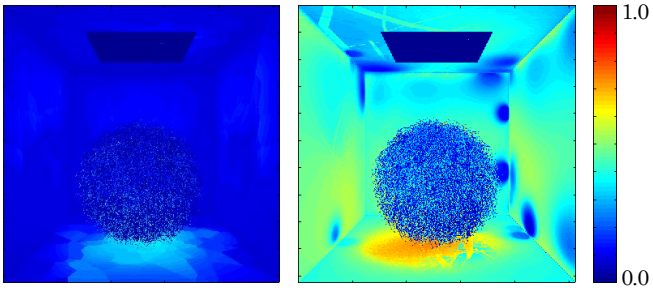

図 4: 提案法（左図）と，Veach の手法（右図）の全 VPL 数に対する可視判定が行われた VPL 数の割合の可視化.

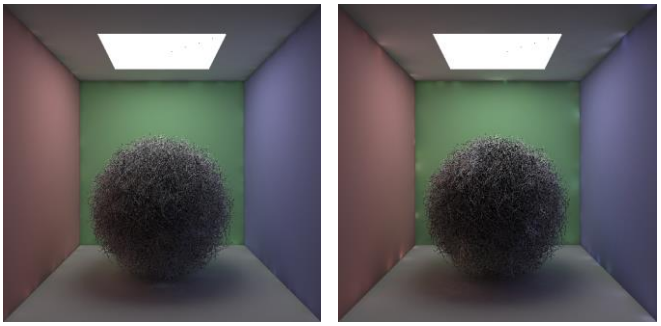

図 5: 提案法（左図）と，全 VPL で可視関数を評価した場合（右図）の同時間レンダリング結果．計算時間と使用した VPL 数はそれぞれ 99 秒と 13k，101 秒と 2k である.

図 2 に(a)提案法，(b)Veach の手法，(c)全 VPL で可視関数を評価した場合（参照画像）の同 VPL 数でレンダリングした結果と，(d)提案法と参照画像の差分画像，(e)Veach の手法と参照画像の差分画像を示す．なお，差分画像は輝度を 100 倍にしたものである．(a)と(c)との差分は輝度を 100 倍にして判別できる程度の差分しかない．また，このときの統計データを表 1，表 2 に示す．提案法の計算時間は，キャッシュ点の生成とレンダリングを合わせた時間である．なお，表 1 の高速化率と表 2 の削減率は，全 VPL で可視関数を評価した場合の計算時間と可視判定回数に対するものである．表 1 より，すべてのシーンで提案手法は Veach の手法より高速にレンダリングできていることが分かる．特に，複雑な形状をしている Hairball シーンのように可視判定に時間がかかるシーンでは，Veach の手法が 2.45 倍の高速化率であるのに対し，提案法では 6.08 倍の高速化を達成した．また，比較的単純なシーンの Sponza シーンにおいても，Veach の手法が 1.47 倍の高速化に対して，提案法では 2.12 倍の高速化を達成した．また，表 2 より，すべてのシーンで提案法は Veach の手法より可視判定回数を削減できていることが分かる．Veach の手法と比較して高速化率が高かった Sanmiguel，Hairball シーンでは，Veach の手法が可視判定回数を約 64%削減しているのに対し，提案法では約 90%削減することができている．

図 3 に，キャッシュ点数別の RMSE と計算時間のグラフを示す．図 3 より，キャッシュ点の数が増加するほど，前処理であるキャッシュ点生成のための計算時間が線形的に増加している．一方，RMSE はキャッシュ点の数が 2k を越える当たりから減少率が小さくなっており，キャッシュ点の数は 2k 程度で十分である．

図 4 に，Hairball シーンでの全 VPL 数に対する可視判定が行われた VPL 数の割合を可視化した画像を示す．Veach の手法では全体的に可視判定が行われた割合が高いのに対し，提案法ではキャッシュ点を使用し，位置および方向に関する可視関数のコヒーレンスを利用しているため，壁や天井のように，複雑な形状の物体がない場所では可視判定が行われた割合が低くなっている．

図 5 に，Hairball シーンでの提案法と全 VPL で可視関数を評価した場合の同時間レンダリングの結果画像を示す．なお，図 5 の画像のレンダリング時のみ，幾何項のクランプの上限値を高く設定している．幾何項のクランプの上限値を高く設定するほど，幾何項の特異点によるアーティファクトが顕著になるが，より真値に近い画像を生成することができる．全 VPL で可視関数を評価した場合は，101 秒の計算時間で約 2k の VPL を扱えるのに対し，提案法では可視関数を確率的に評価するため，99 秒の計算時間で約 13k の VPL を扱うことができる．全 VPL で可視関数を評価した場合では，全てのシェーディング点で同一の VPL を使用するため，画像にノイズが発生しないが，使用する VPL の数が少ないため，VPL を用いたレンダリングに特有の幾何項の特異点によるアーティファクトが表れている．一方，提案法では，多くの VPL を使用しているため，アーティファクトが緩和されている．また，可視関数の

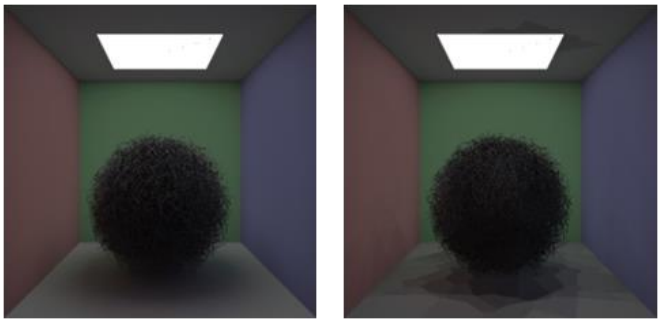

図 6: 提案法（左図）と可視関数v_iにキャッシュ点からの補間値q_iを用いた場合（右図）のレンダリング結果.

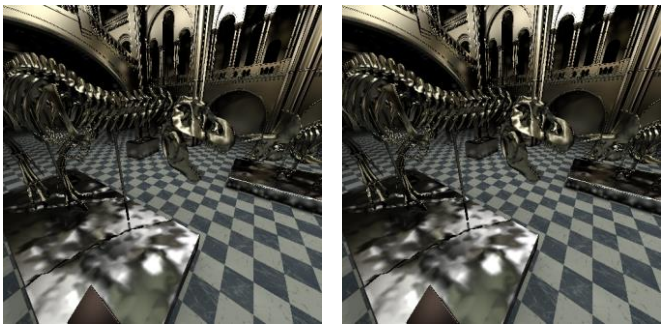

図 7: 提案法（左図）と，全 VPL で可視関数を評価した場合（右図）の光沢反射モデルのレンダリング結果．使用した VPL 数は 300k で，計算時間はそれぞれ 1,468 秒と 3,822 秒である.

評価を省略する確率に, 画像生成の効率を最大とする確率を用いることで，画像に目に見えるほどのノイズはない.

図 6 に，提案法によるレンダリング結果と，可視関数v_iにキャッシュ点からの補間値q_iを用いた場合（つまり，$L_o = \Sigma_{i=1}^{N} t_i q_i$）のレンダリング結果を示す．図 6 より，可視関数v_iの代わりにキャッシュ点からの補間値q_iを用いた場合では, 近傍のキャッシュ点が異なる場所（床，光源付近の天井）で，輝度の変化が不連続になっている．提案法では，キャッシュ点からの補間値q_iを可視関数として使用せず，可視関数の推定に使用しているため，輝度の変化が不連続な場所はない.

図 7 に，Museum シーンでの，提案法と全 VPL で可視関数を評価した場合の光沢反射モデルのレンダリング結果を示す. 使用した VPL 数は 300k で，提案法の計算時間は 1,468 秒，全 VPL で可視関数を評価した場合の計算時間は 3,822 秒である．提案法では，全 VPL で可視関数を評価した場合と同程度の画質で半分の時間でレンダリングできている．提案法，全 VPL で可視関数を評価した場合の両方で，VPL を用いたレンダリングに光沢反射モデルを適用した際に生じる斑点状の模様が現れている. このアーティファクトを緩和するには，より多くの VPL を用いる必要があり，提案法では，同時間でより多くの VPL を扱うことができる.

6 まとめ

本稿では, 確率的な手法を用いて可視判定を行う回数を削減する手法を提案した．また，可視判定を省略する確率と確率的評価による分散の関係を考慮し, 画像生成の効率を最大とする確率の導出を行った．多光源レンダリング法において，Veach の手法より効率的にレンダリングを行えることを示した．特に，可視判定のコストが高いシーンで大きな高速化率を得た．今後の課題として，GPU 実装による高速化，シェーディング点と VPL のクラスタリングによる高速化が挙げられる.

参考文献

[1] A. Keller, Instant Radiosity, Proc. of SIGGRAPH '97, 49-56, 1997.

[2] C. Dachsbacher, J. Krivanek, M. Hasan, A. Arbree, B. Walter, and J. Novak, Scalable Realistic Rendering with Many-light Methods, Eurographics STAR, 2013.

[3] B. Walter, S. Fernandez, A. Arbree, K. Bala, M. Donikian, and D. Greenberg, Lightcuts: A Scalable Approach to Illumination, ACM TOG, Vol. 24, No. 3, pp. 1098-1107, 2005.

[4] M. Hasan, F. Pellacini, and K. Bala, Matrix Row-Column Sampling for the Many-Light Problem, ACM TOG, Vol. 26, No. 3, pp. 26:1-26:10, 2007.

[5] J. Ou, and F. Pellacini, LightSlice; Matrix Slice Sampling for the Many-Lights Problem, ACM TOG, Vol. 30, No. 6, pp. 179:1-179:8, 2011.

[6] I. Georgiev, J. Křivánek, S. Popov, and P. Slusallek, Importance Caching for Complex Illumination, Computer Graphics Forum Vol. 31, No. 2, pp. 701-710, 2012.

[7] P. Clarberg, and T. Akenine-Moller, Exploiting Visibility Correlation in Direct Illumination. Computer Graphics Forum, Vol. 27, No. 4, pp. 1125-1136, 2008.

[8] S. Popov, I. Georgiev, P. Slusallek, and C. Dachsbacher, Adaptive Quantization Visibility Caching, Computer Graphics Forum, Vol. 32, No. 2, pp. 399-408, 2013.

[9] J. Ulbrich, J. Novak, H. Rehfeld, and C. Dachsbacher, Progressive Visibility Caching for Fast Indirect Illumination, Proc. of International Workshop on Vision, Modeling, Visualization, pp. 203-210, 2013.

[10] N. Billen, B. Engelen, A. Lagae, and P. Dutré, Probabilistic Visibility Evaluation for Direct Illumination. Computer Graphics Forum, Vol. 32, No. 4, pp. 39-47, 2013.

[11] E. Veach, Robust Monte Carlo Methods for Light Transport Simulation. PhD thesis, Stanford University, 1998.