多重スケール環境マップを用いた粗さが不均一な材質への写り込みの高速レンダリング

Fast Rendering of Reflections in Non-uniform Surfaces Using a Multi-Scale Environment Map

奥村文洋†，町田貴史††，正会員 横矢直和†

Bunyou Okumura†, Takashi Machida†† and Naokazu Yokoya†

Abstract A technique is described for rendering in real time reflections on surfaces with non-uniform roughness, which is difficult to do with conventional methods. It combines off-line processing with hardware rendering. First, a multiscale environment map is created of the rough surfaces. Next, the pixels that would significantly affect the rendering results in an omnidirectional image are filtered out. Finally, the reflections on the surfaces are rendered in real time using the map to reflect the effects of the roughness. Experiments demonstrated that the proposed method can render surfaces with non-uniform roughness sufficiently accurately in real time.

キーワード：レンダリング，環境マッピング，写り込み，Phongモデル，全方位実写画像

1. まえがき

近年，コンピュータグラフィックス(CG)の分野において，写実的な画像を生成するための研究が多数なされている[1)~3)]．なかでも，仮想物体の写実性を高める要素である写り込みを，高速にレンダリングする手法として環境マッピング[4)]が提案された．しかし，環境マッピングでは材質の反射特性を考慮できないという問題があった．このため，環境マッピングに対してさまざまな改良が行われた[5)~8)]．これらの研究では，環境マッピングを拡張することで，反射特性を考慮した写り込みのレンダリングを可能としている．Millerら[5)]は，実時間レンダリングは困難なものの，鏡のような材質ではなく，粗さを持った材質に対する写り込みのレンダリング手法を提案している．また，Heidrichら[7)]は，Millerの手法を改良して粗さを持った物体の写り込みを実時間で表現している．また，別のアプローチではCabralら[6)]は，想定した材質から成る球状の実物体に環境を写り込ませ，その球体をカメラで撮影し，テクスチャマッピングにより実時間レンダリングを可能としている．Kautzら[8)]は，写り込む情報をあらかじめフィルタ処理で計算しておくことで，単一の粗さの写り込みを実時間で表現可能としている．以上のいずれの手法においても，単一の反射特性を想定しているため，反射特性が不均一な材質から構成される物体をレンダリングする場合，材質毎に環境マップを用意し，それらをすべて用いてレンダリングする必要があり，膨大な計算時間を要する．そのため，仮想環境内でのウォークスルーなどの高い実時間性を要求するアプリケーションでは，その表現が限定されるという問題がある．

本研究では，反射特性が不均一な材質，特に粗さが不均一な材質に対して写り込みの高速レンダリング手法を提案する．具体的には，オフライン処理として環境マップに対してあらかじめフィルタ処理を行い，複数の材質の粗さに応じた環境マップ(多重スケール環境マップ)を作成し，それらをボクセルデータとしてまとめて保持する．そして，計算された多重スケール環境マップを，3Dテクスチャとして利用することで，写り込みをテクスチャマッピングによって高速にレンダリングする．このとき，粗さに応じたテクスチャ座標を設定することで，粗さが不均一な材質であっても，写り込みを一度にレンダリングすることが可能である．また，グラフィックスハードウェアの機能を利用することで，実時間レンダリングが可能となる．これにより，材質が均一な場合は，従来法に比べ前処理のフィルタ計算による計算量が大きくなるものの，仮想環境中で様々な材質からなる物体に対する写り込みを実時間でレンダ

2003年3月3日受付，2003年7月17日最終受付，2003年8月12日採録
† 奈良先端科学技術大学院大学 情報科学研究科
（〒630-0192 生駒市高山町 8916-5, 0743-72-5296）
†† 大阪大学 サイバーメディアセンター
（〒560-0043 豊中市待兼山町 1-32, 06-6850-6824）
† Graduate School of Information Science, Nara Institute of Science and Technology
(8916-5 Takayama, Ikoma, Nara, 630-0101 Japan)
†† Cybermedia Center, Osaka Univercity
(1-32 Kaneyama, Tonaka, Osaka, 560-0043 Japan)

リングでき，仮想環境内におけるウォークスルーなどのアプリケーションにおいて，より写実性の高い表現が可能となる．

以下，2章において多重スケール環境マップの詳細とその計算方法，写り込みのレンダリング法，さらに計算の高速化手法について述べる．3章では高速化の影響の検証，Phongモデルとの比較実験を示し，提案手法によるレンダリング結果を示す．このとき，多重スケール環境マップをあらかじめ多数作成しておくことにより，環境マップとして動画を利用したレンダリング結果を示す．そして最後に考察とまとめを述べる．

2. 多重スケール環境マップを用いた写り込みのレンダリング

2.1 材質の反射モデル

本研究では，材質の反射モデルとしてPhongモデル[9]を用いる．Phongモデルは，材質の粗さを表現可能な照明モデルであり，図1のように物体表面上のある点$\mathbf{x}$に対して，視点方向を$\vec{v}$，光源の方向を$\vec{l}$，物体表面の法線を$\vec{n}$，視点ベクトルの正反射ベクトルを$\vec{r}$，$\vec{r}$と$\vec{l}$のなす角をθ，$\vec{n}$と$\vec{l}$のなす角をϕとしたとき，写り込みの輝度が$\cos^{\Lambda}(\theta)$となる照明モデルである．なお，各ベクトルは単位ベクトルとする．Λは粗さ係数となり，Λが大きいほど表面が滑らかな材質となる．

一般に，Phongモデルによって環境内に配置された物体に対する写り込みをレンダリングする場合，入力となる環境をΩ，Ω上の光源iの輝度をL_i，物体固有の表面反射特性を現すユーザが設定可能な係数を$K_r (0 \leqq K_r \leqq 1)$とすると輝度値の計算は式(1)を用いて行われる．

$$L(\mathbf{x};\vec{\mathbf{v}},\vec{\mathbf{n}}) = K_r \int_{\Omega} f(\cos(\theta),\cos(\phi),\Lambda) L_i \, d\vec{l} \qquad (1)$$

ただし，

$$f(\cos(\theta),\cos(\phi),\Lambda) = \begin{cases} \cos^{\Lambda}(\theta) & ;\ \text{if } \cos(\theta) \geq 0 \\ & \quad \text{and } \cos(\phi) \geq 0 \\ 0 & ;\ \text{otherwise} \end{cases} \qquad (2)$$

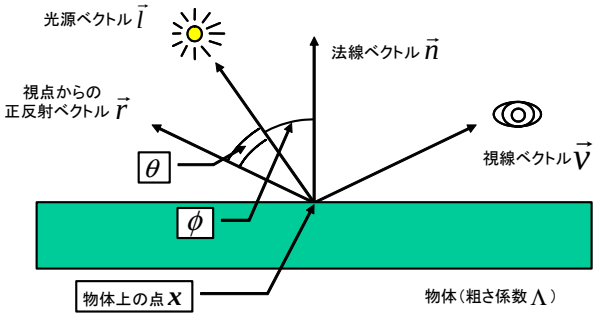


図 1 物体と視点と光源の関係
Illustration of surface reflection.

ここで，関数$f(\cos(\theta),\cos(\phi),\Lambda)$はPhongモデルに基づいた反射輝度を決定する式となっており，θによって写り込みの輝度強度が決定され，ϕによって光源が物体を照らすかどうかの判定を行っている．

2.2 多重スケール環境マップ

Phongモデルを用いて自然な照明計算を行うためには，Ω上に多数の光源を配置しなければならない．しかし，実時間でのレンダリングを行う場合，多数の光源を用いた照明計算は計算時間の観点から困難である．

そこで本研究では式(2)に着目する．一般に表面が粗い反射特性を持つ材質であっても，Λはあまり小さくならないことが経験的に知られており，θがある程度大きくなると$\cos^{\Lambda}(\theta)$の値は非常に小さくなる．そこで，写り込みの計算を行う場合に，光源が物体を照らすかどうかの判定，すなわち，ϕによる判定を省くことで式(1)を式(3)のように変形する．

$$L(\mathbf{x};\vec{\mathbf{v}},\vec{\mathbf{n}}) = K_r \int_{\Omega} f'(\cos(\theta),\Lambda) L_i \, d\vec{l} \qquad (3)$$

$$= K_r F(\vec{r},\Lambda;\Omega) \qquad (4)$$

ただし，

$$f'(\cos(\theta),\Lambda) = \begin{cases} \cos^{\Lambda}(\theta) & ;\ \text{if } \cos(\theta) \geq 0 \\ 0 & ;\ \text{otherwise} \end{cases} \qquad (5)$$

この変形によって，物体に隠されている光源からの影響を受けるようになるため，表面が粗い材質(Λが小さい場合)を物体に対して浅い角度で観測する場合（$\vec{n}$と$\vec{v}$のなす角が$\pi/2$に近い場合）に誤差が生じるが，ここでは，そのような状況の影響は無視できると仮定する．

また，式(3)の右辺を式(4)のように表すと，Fは表面の粗さがΛであり，視線からの正反射ベクトルが$\vec{r}$となる物体上の点を観測した場合の写り込みの輝度値を得るための関数となる．このとき，Fは環境Ωにのみ依存するため，環境に変化がないと仮定すれば，あらかじめ計算することができ，正反射ベクトルの方向$\vec{r}$と表面粗さ係数Λの3次元のパラメータで記述可能なボクセル空間で表現することができる．そこで，写り込みの輝度値をボクセルデータとして記録しておき，レンダリング時にボクセルを参照することで，高速なレンダリングが可能となる．以下，この写り込み情報を多重スケール環境マップと呼ぶ．

（1） 材質の粗さに応じた写り込み画像のボクセルデータ化

保持される写り込み情報のうち，表面が粗い材質に対する写り込みは，強いボケが生じていると考えられるため，写り込み情報を保持するために高い解像度を割り当てることは無駄である．そこで，表面が粗い材質に対応する写り込み情報は，ボクセル空間の中心付近に，表面が滑らかな材質に対応する写り込み情報は，ボクセル空間の外周付近に格納する（図2参照）．こうすることで，粗い材質に対しては低解像度で，滑らかな材質に対しては，高解像度で写

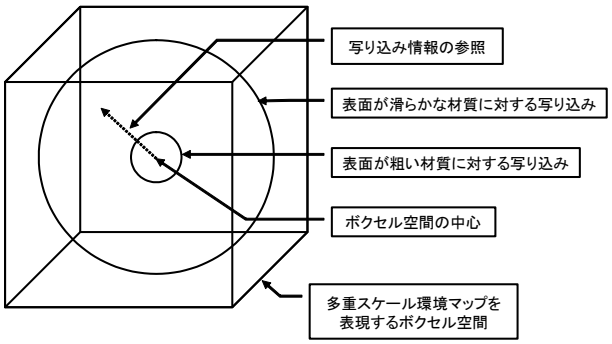


図 2 ボクセル空間内に配置される写り込み情報
Multi-scale environment map represented in voxel space.

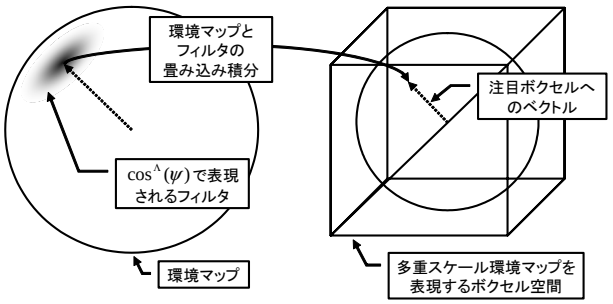


図 3 多重スケール環境マップの計算
Calculation of multi-scale environment map.

り込みの情報を保持することができる．

多重スケール環境マップを表現するボクセル空間の中心からの方向が反射ベクトル，ボクセル空間の中心からの距離が材質の滑らかさに対応し，写り込み情報の参照が容易になる．つまり，ボクセル空間の中心からあるボクセル$\mathbf{t}$へのベクトルを$\vec{t}$とすると，以下の関係が成立する．

$$\vec{t} = \vec{r} \cdot \lambda \tag{6}$$

$$\Lambda = g(|\vec{t}|) = g(\lambda) \tag{7}$$

ただし，λは提案手法における粗さの表現であり，Phongモデルにおける粗さΛとの対応は関数gによって与えられる．この関数については，(2)で詳しく述べる．

入力となる単位球面上に定義された環境マップをΩ，環境マップの中心から環境マップ上のある画素へのベクトルを$\vec{\omega}$とすると，多重スケール環境マップを計算するための式は以下の通りである．

$$F(\mathbf{t};\vec{\mathbf{t}}) = \sum_{\vec{\omega}:0\leqq\vec{t}\cdot\vec{\omega}} \Omega(\vec{\omega}) \cdot w(\vec{t},\vec{\omega}) \tag{8}$$

ただし，

$$w(\vec{t},\vec{\omega}) = \frac{(\vec{t}\cdot\vec{\omega})^{\Lambda}}{\sum_{\vec{\rho}:0\leqq\vec{t}\cdot\vec{\rho}}(\vec{t}\cdot\vec{\rho})^{\Lambda}} \tag{9}$$

$$\Lambda = g(\lambda) \tag{10}$$

各ボクセルの値は，入力とする環境マップの各画素を色つきの点光源とみなし，各光源に対してPhongモデルによる重み$w(\vec{t},\vec{\omega})$との重み付き加算を行い，その総和をとることで得られる．この計算は$\vec{t}$と$\vec{\omega}$のなす角をψとおけば，入力となる環境マップに対して$\cos^{\Lambda}(\psi)$で表現されるフィルタの畳込み積分を行うのと等価である（図3参照）．

（2） 材質の粗さの表現

多重スケール環境マップでは，ボクセル空間の中心からの距離が材質の粗さに対応しているため，ボクセル空間内で粗さが等しい点を選ぶと，ある半径の球面を構成する．このとき球面を構成するボクセルの数によって，ある粗さの写り込みを保持するのに用いられるボクセル数，すなわち，ある粗さの環境マップを保持する解像度が決定される．ここで，滑らかな材質に対する写り込みを保持する場合に，解像度が低いと写り込みを正確に保持できない問題が発生する．

そこで，材質への写り込みがボクセルデータとして正しく保持できる条件が必要となる．これは，ある粗さの写り込みを保持する環境マップの空間分解能が$\cos^{\Lambda}(\psi)$で示されるフィルタのサイズを上回っていればよい．本研究では簡単のため，中心からの距離が同じである隣り合った二つのボクセルへのベクトルのなす角をψ_{voxel}，ボクセルデータの輝度値の分解能をI，ボクセル空間の解像度をsとした時，条件式(11)を満たすΛを表現可能な粗さとする．

$$\frac{1}{I} \geqq \cos^{\Lambda}(\psi_{voxel}) \tag{11}$$

ただし，

$$\psi_{voxel} = \arctan(u/(\lambda \times s)) \tag{12}$$

ここで，uはフィルタの畳込み積分のためのウィンドウ幅を表している．以上より式(7)の$g(\lambda)$は

$$\Lambda = g(\lambda) = \frac{-\log(I)}{\log(\cos(\arctan(u/(\lambda \times s))))} \tag{13}$$

となる．なお，uに関しては次の方法であらかじめ算出しておく．まず，$\lambda = 1.0$が提案手法で鏡のような材質を表現できるとする．これに対して，Phongレンダリングにおいて同様の材質を表現すると$\Lambda = 5000$となることを確認している．そこで，式(13)にそれらの値を代入することでuを算出する．この結果$u = 6.2$となり，以降ではこの値を用いる．

2.3 多重スケール環境マップを用いたレンダリング

写り込みのレンダリングは，多重スケール環境マップとして計算されたボクセルデータを3Dテクスチャとして，テクスチャマッピングすることで行う．図4は，写り込みをレンダリングする際のテクスチャ座標の決定方法を示している．ボクセルデータのサンプリングに用いられるテクスチャ座標$\vec{t}$は，ユーザが設定する粗さ係数$\lambda(0 < \lambda \leqq 1)$と正規化された視点からの反射ベクトル$\vec{r}$より式(6)で計算される．なお，$\lambda$の値は0に近いほど表面の材質が粗いことを表し，1に近いほど表面が鏡のように滑らかな材質を表している．この計算は，実時間で計算可能であるため，ユーザは実時間で材質の粗さを変更することが可能である．

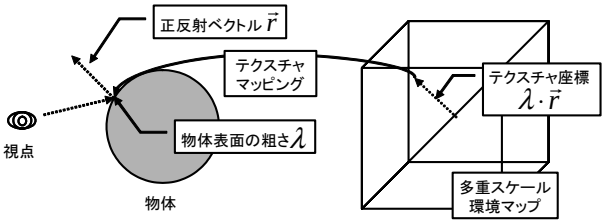


図 4 テクスチャマッピングによる写り込みのレンダリング
Texture mapping using a multi-scale environment map.

このとき，粗さ係数を描画するポリゴンの頂点単位に設定することで，粗さが一様でない材質に対する写り込みを一度にレンダリングすることが可能である．また，正反射ベクトルの計算には，近年のグラフィックハードウェアの機能であるテクスチャ座標生成を用い，さらに，ハードウェアの3Dテクスチャマッピング機能を利用することで，高速なレンダリングを行う．

2.4 多重スケール環境マップ作成の高速化

式(8)は畳込み積分を行う計算であるため，計算量は環境マップ上の画素数と多重スケール環境マップのボクセル数に依存し，単純に実行すれば膨大な計算量となる．そこで，入力となる環境マップに解像度の異なる複数の画像を用いることで，計算の簡略化を行い，さらに重みをテーブル化することで，レンダリング結果に影響の少ない計算を省く．

（1） ピラミッド画像と測地ドームの利用

多重スケール環境マップにおいて，ボクセル空間の中心付近の情報，すなわち表面が粗い材質に対応する写り込み情報は，計算結果を格納する際の解像度が低く割り当てられている．しかし，式(8)の計算ではボクセル空間の中心付近に存在するボクセルの輝度値を決定する際にも，入力となる環境マップの画素を利用して計算している．これは，粗い材質に対する写り込みを計算するために，高い解像度の環境マップ使用することになり，計算が膨大になる原因である．

そこで，表面が粗い材質に対応する写り込み情報の計算には，あらかじめ線形フィルタで縮小された環境マップ（ピラミッド型画像）を利用して，式(8)の計算を行う(図5参照)．これによって，結果に大きな影響を与えることなく，計算量を削減することができる．また本手法では，入力となる環境マップから細分化された測地ドームを用いて再サンプリングを行う．これにより，入力となる環境マップの形式に依存することがなくなり，さらに環境マップの形式による解像度の偏りがなくなる．

（2） 重みテーブルの利用

提案手法では，さらに計算の高速化のために式(8)の重みをあらかじめ計算することでテーブル化を行った．このとき，閾値thを用いて$\cos^{\Lambda}(\psi) > th$を満たす重みのみをテーブルに記録する．これにより，計算結果に大きな影響を与える環境マップの画素のみを計算対象とすることができ，計算量の削減が行える．

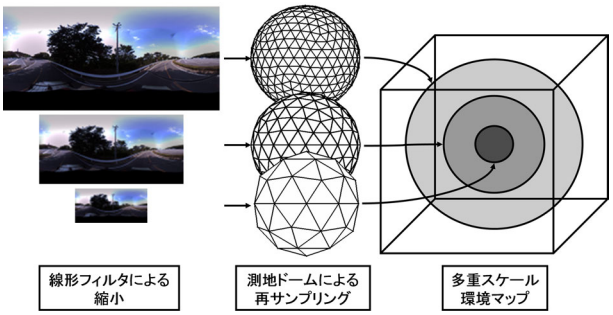


図 5 複数の解像度の画像を用いたフィルタ計算
Generation of a multi-scale environment map using multi-scale images.

表 1 重みテーブルの例
Table of weighting coefficients

対象ボクセル	環境マップ上の参照画素と重みのリスト
$F(\vec{t}_0)$	$(\vec{\omega}_{0,0}, w_{0,0}), \quad \cdots \quad, (\vec{\omega}_{0,M_0}, w_{0,M_0})$
$F(\vec{t}_1)$	$(\vec{\omega}_{1,0}, w_{1,0}), \quad \cdots \quad, (\vec{\omega}_{1,M_1}, w_{1,M_1})$
$\vdots$	$\vdots$
$F(\vec{t}_N)$	$(\vec{\omega}_{N,0}, w_{N,0}), \quad \cdots \quad, (\vec{\omega}_{N,M_N}, w_{N,M_N})$

表1は，ボクセルの総数はN個，i番目のボクセルを計算する時に計算対象となる環境マップの画素数はM_i個とした場合の，重みテーブルに保存されるデータ形式を示す．計算結果を格納するボクセルに対して，参照すべき環境マップ上の画素と重みの組を表現している．各ボクセルの値は，このテーブルを用いて式(14)を計算することで決定される．

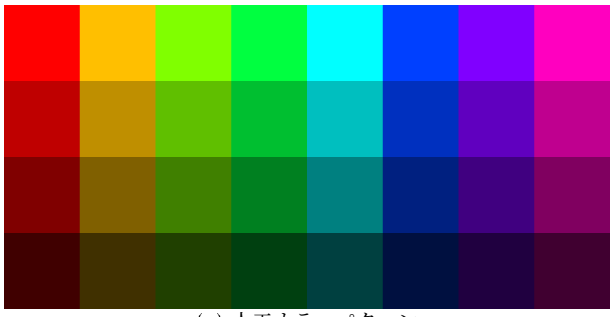
$$F(\vec{t}_i) = \sum_{j=0}^{M_i} \Omega(\vec{\omega}_{i,j}) \times w_{i,j} \qquad (14)$$

3. 実 験

提案手法の妥当性を検証するための実験を行った．計算にはPentium4 Xeon(1.7GHz Dual，メモリ 2GByte)，グラフィックカードとしてNVIDIA社GeForce3(VRAM 64MByte)を搭載した計算機を用いた．また，すべての実験において表面反射係数K_rは$(R,G,B) = (1.0, 1.0, 1.0)$とした．

3.1 近似計算による高速化の影響

2.4節において，多重スケール環境マップの計算を高速化する手法を示した．本節では，高速化を行った手法と高速化を行わなかった手法を比較することで高速化手法の影響を検証する．本実験では図6に示す環境マップを用いた．図6(a)の画像は横方向に色相，縦方向に彩度が変化する人工カラーパターンである．図6(b)の画像は全方位マルチカメラシステムLadybug[10]を用いて撮影された高山サイエンスプラザ内の全天球パノラマ画像である．両画像の解像度は512×256であり，高速化の際には64×32までの4段階の縮小を行い，測地ドームも頂点数51842から812ま



(a) 人工カラーパターン

(b) 高山サイエンスプラザの全方位画像

図 6 実験に用いた二つの環境マップ
Environment maps used in experiments

表 2 高速化に伴うボクセルの輝度値の誤差(256 階調)
Intensity differences caused by decreasing computation time.

入力画像	閾値(th)	平均(r, g, b)	分散(r, g, b)
図6(a)	0.001	(0.1, 0.2, 0.3)	(0.01, 0.01, 0.01)
	0.005	(1.9, 1.9, 2.0)	(0.03, 0.04, 0.05)
	0.01	(3.8, 3.8, 4.0)	(0.13, 0.13, 0.14)
	0.05	(19, 19, 19)	(3.24, 3.35, 3.37)
	0.1	(38, 38, 38)	(12, 13, 13)
図6(b)	0.001	(1.0, 1.0, 1.0)	(0.02, 0.02, 0.02)
	0.005	(2.0, 2.0, 2.1)	(0.04, 0.05, 0.05)
	0.01	(3.6, 3.6, 3.6)	(0.12, 0.11, 0.13)
	0.05	(17.8, 17.4, 19.2)	(2.6, 2.6, 3.0)
	0.1	(35, 35, 38)	(10, 10, 12)

表 3 計算時間の比較
Comparison of computation time.

計算手法	閾値(th)	テーブル作成時間 [h:m:s]	フィルタ計算時間 [h:m:s]	合計時間 [h:m:s]	テーブルの大きさ [MByte]
高速化なし	—	—	73:51:54	73:51:54	—
高速化あり	0.001	21:23:41	0:00:09	21:23:50	459
	0.005		0:00:09	21:23:50	258
	0.01		0:00:08	21:23:49	216
	0.05		0:00:08	21:23:49	116
	0.1		0:00:07	21:23:48	74.1

での4段階を用意した．さらに，高速化のときに用いた閾値は$th = 0.001, 0.005, 0.01, 0.05, 0.1$の5通りで計算を行った．なお，生成した多重スケール環境マップの解像度は$128 \times 128 \times 128$であり，並列処理は行っていない．

表2に，図6(a),(b)それぞれの環境マップを用いた際の閾値，高速化を行わない場合を真値とした，各ボクセルの輝度値の誤差の平均および分散の関係を示す．表2より，閾値に0.001程度の値を設定した場合で，輝度値の誤差は最大でも1.0程度に収まっている．一般の表示装置では，輝度値の差が256階調で1未満である場合はその違いを表示することができず，さらに1程度の輝度値の違いは，人間がその差を知覚することは難しい．よって，高速化に伴って生じた計算誤差は小さく，高速化手法の悪影響は小さいと言える．なお，提案手法におけるフィルタ処理は入力画像の特徴（エッジなど）に強い影響を与えるため，エッジ付近で各RGB値が異なる場合，それぞれのチャネルで生じる誤差が異なる結果となる．

また，それぞれの場合の計算時間と計算に使用した重みテーブルの大きさを表3に示す．表3より高速化により計算が約3.5倍程度速くなっていることがわかる．さらに，重みデータのテーブル化を行うことで，複数の多重スケール環境マップを作成する場合にもフィルタ計算のみを行えばよく，多重スケール環境マップの作成に有利となっている．なお，テーブルの大きさは最大でも500MByte程度であり，近年の計算機では問題にならないといえる．

3.2 Phongモデルとの比較

本節では，提案手法によるレンダリング結果が式(1)のPhongモデルによるレンダリング結果と比べてどの程度の差があるかの検証実験の結果を示す．実験方法として，3.1節で用いた図6の二つの環境マップを用い，ティーポットに対して提案手法とPhongモデルによるレンダリングを行った．粗さは$\lambda = 1.0, 0.75, 0.5, 0.25$，不均一なものの合計5通りである．不均一な粗さの場合は，まずティーポットを正面から見て，それを囲む3次元の直方体を考える．ここで，直方体の下半分の領域に入る頂点は表面が粗いものとして，表面粗さの設定を$\lambda = 0.1$とし，上半分の領域に入る頂点は表面が滑らかなものとして，表面粗さの設定を$\lambda = 1.0$とした．なお，領域の境界に位置する頂点に関しては，上半分の領域とした．そして，Phongモデルによるレンダリングを真値として結果画像の輝度値の誤差の平均，分散を示し，計算時間についても比較する．なお，多重スケール環境マップの解像度は前節と同様に$128 \times 128 \times 128$とし，高速化の際の閾値は$th$=0.001とした．また，Phongモデルによるレンダリングは提案手法で指定した粗さ係数λと対応する粗さ係数を与えた．

輝度値の比較を行った結果を表4に示す．レンダリングに用いた物体はティーポットであり，二つの表はそれぞれ，人工的なパターン（図6(a)）と実写画像（図6(b)）の写り込みに対して，Phongモデルによるレンダリング結果と提案手法によるレンダリング結果の画像間の誤差の平均と分散を表している．それぞれの誤差に関しては，材質の粗さが粗いほど誤差が大きくなる傾向にあるが，Phongモデルと比較して，256階調で輝度値にして最大でも1.5程度となっており，式(13)が提案手法における粗さ表現と，Phongモデルにおける粗さ表現との対応をうまく近似できていることが確認できる．

(a) Phongモデルによる結果　(b) 提案手法(高速化あり)　(c) Phongモデルとの差分画像

図 7　レンダリング結果の比較　(不均一な粗さを持つティーポット，実写画像使用)
Comparison of rendring results of a virtual teapot with non-uniform surface roughness.

表 4　Phongモデルによるレンダリング結果との比較
Comparison with Phong's model-based rendering in terms of computed intensities.

(a) ティーポットを用いた比較，図6(a)使用

粗さ	誤差の平均	誤差の分散
$\lambda = 1.0$	(0.35, 0.35, 0.35)	(0.02, 0.02, 0.02)
$\lambda = 0.75$	(0.48, 0.48, 0.48)	(0.02, 0.02, 0.02)
$\lambda = 0.5$	(0.65, 0.64, 0.64)	(0.03, 0.03, 0.03)
$\lambda = 0.25$	(0.90, 0.91, 0.90)	(0.04, 0.04, 0.04)
不均一	(0.56, 0.56, 0.56)	(0.03, 0.02, 0.03)

(b) ティーポットを用いた比較，図6(b)使用

粗さ	誤差の平均	誤差の分散
$\lambda = 1.0$	(0.73, 0.73, 0.73)	(0.03, 0.03, 0.03)
$\lambda = 0.75$	(0.86, 0.86, 0.86)	(0.02, 0.02, 0.02)
$\lambda = 0.5$	(0.91, 0.91, 0.91)	(0.02, 0.02, 0.02)
$\lambda = 0.25$	(1.4, 1.5, 1.5)	(0.02, 0.03, 0.03)
不均一	(1.2, 1.2, 1.2)	(0.04, 0.04, 0.04)

図7は，図6(b)を用いて表面の材質粗さが不均一なティーポットに対してレンダリングを行った結果である．Phongモデルによるレンダリング結果が(a)，提案手法によるレンダリング結果が(b)，(a)と(b)の差分画像が(c)である．なお，(a)については，一番解像度の高い測地ドーム上の各頂点に対応する環境マップの各画素の色をつけた点光源があると仮定してレンダリングを行っている．また，(c)についてはその差をわかりやすくするため，輝度値を4倍にスケーリングしている．図7からもわかるように，画像で確認しても違いがほとんど感じられない．このことからも提案手法によるレンダリング結果は，Phongモデルをうまく近似しており，充分に実用的な写実性をもっていると考えられる．なお，誤差が大きくなる原因については，多重スケール環境マップに粗さに応じた写り込み情報を保持する際の解像度不足が考えられる．また，誤差が生じている部分に関しては，図7(c)において白い円で囲んでいる部分のように，視線と物体法線が90°に近くなる輪郭部分や，写り込んだ物体のエッジ付近で誤差が見られる．前者は式(3)を用いたPhongモデルに対する近似による影響，

表 5　Phongモデルとの計算時間の比較
Comparison with Phong's model-based rendering in terms of computation time.

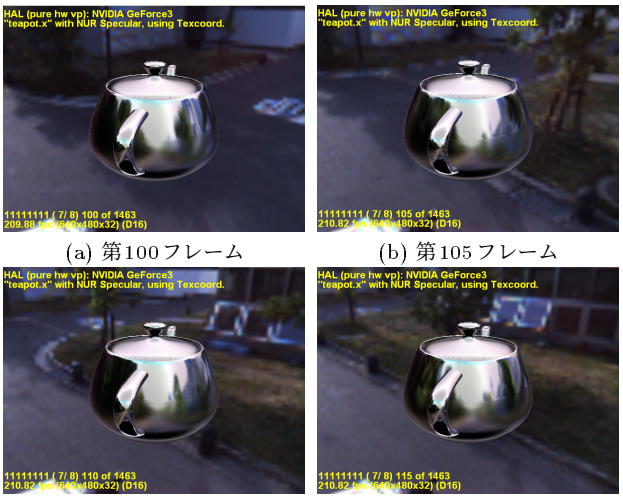
手法	多重スケール環境マップの作成時間 [h:m:s]	レンダリング時間[h:m:s]
提案手法(高速化なし)	73:51:54	< 0:00:0.017(60fps)
提案手法(高速化あり)	21:23:49	< 0:00:0.017(60fps)
Phongモデル	—	1:03:05

後者は多重スケール環境マップの中心付近の情報を用いてテクスチャマッピングする際の線形フィルタによる影響が強く出たものと考えられる．また，ティーポットの上部と下部の境界付近が線形補間されるため，その境界がやや不自然となる場合があるが，これは現状ではグラフィックスハードウェアの機能を有効利用するために，物体の頂点単位での計算を行っているためである．将来，グラフィックスハードウェアの性能の向上により，画素単位での処理が可能となれば，より細かい単位での描画が可能になると考えられる．

計算時間についての比較を表5に示す．提案手法では，前処理に21時間程度かかるが，一度多重スケール環境マップを作成すれば，60fps以上で実時間レンダリングを行うことが可能である．対してPhongモデルによるレンダリングでは，1フレームあたり1時間程度を要するため，写り込みの高速なレンダリングは困難である．なお，これらの計算時間は物体の材質には依存しない．

3.3　環境マップに動画像を用いた写り込みのレンダリング

全方位マルチカメラシステムLadybug[10]を車に搭載して撮影された全周パノラマ動画像を用いて，さまざまなシーン内での仮想物体に対する写り込みのレンダリングを行った実験結果を示す．シーンを動画に拡張する際には，多重スケール環境マップの計算をあらかじめ行い，時系列環境マップを蓄積した．本実験で用いた全周パノラマ動画像は15fpsでキャプチャされ，長さ695フレーム(46秒程度)である．仮想物体として，3.2節で利用したティーポット用いた．なお，多重スケール環境マップの解像度は，各フレーム$64 \times 64 \times 64$，背景画像となる環境マップの解像



(a) 第100フレーム　(b) 第105フレーム

(c) 第110フレーム　(d) 第115フレーム

図 8　ティーポットへの写り込みのレンダリング
Rendering of reflections in a virtual teapot using a dynamic environment map.

度は512 × 256であり，総データ量は1GByte程度である．また実行の際には2CPU を使用した．

図8に，粗さが不均一なティーポットに対する写り込みのレンダリング結果を示す．粗さが不均一な材質を持つ仮想物体に対して，粗さに応じた写り込みがレンダリングされていることが確認できる．また，動画像の更新はキャプチャ時と同じ15fpsで行い，視点変更は60fps以上で操作可能であった．なお，695フレームの多重スケール環境マップを作成するのに要した時間は，重みテーブルの作成に2時間17分，フィルタ計算に45分（1フレームあたり約4秒）であった．

4. むすび

本論文では，あらかじめ粗さに応じた写り込み情報を計算することで，粗さが不均一である材質に対する写り込みを高速にレンダリングする手法を示し，さらに多重スケール環境マップの作成を高速化する手法を示した．実験では，高速化手法が妥当であることを確認し，さらに提案手法とPhongモデルによるレンダリング結果を比較することで，輝度値の誤差が256階調で平均1程度あることを確認した．そして，さまざまな実シーン内での写り込みの実時間レンダリングが可能であることを確認した．

今後の課題としては，提案手法のPhongモデルの粗さ表現の近似の誤差を少なくすることや，より写実性の高い表現を可能とするために，画素単位での計算を可能とすることが挙げられる．また，多重スケール環境マップを用いて，詳細な画像の写り込みをレンダリングする際には，ボクセルデータ形式のメモリ消費量の大きさが問題として挙げられる．そこで，多重スケール環境マップに対して，効果的な圧縮を行うことや，他の環境マッピングと組合せることで，より鮮明な写り込みのレンダリングを行う予定である．

〔文　献〕

1) E. Lindholm, M. J. Kilgard and H. Moreton: "A User-Programable Vertex Engine," Proc. SIGGRAPH '01, pp. 149-158, 2001.
2) H. W. Jansen, S. R. Marschner, M. Levoy and P. Hanrahan: "A Practical Model for Subsurface Light Transport," Proc. SIGGRAPH '01, pp. 511-518, 2001.
3) 奥村 文洋，町田 貴史，横矢 直和: "実写に基づく全周多重スケール環境マップを用いた不均一物体への写りこみの高速レンダリング"，信学技報，PRMU2002-137, 2002.
4) J. Blinn and M. Newell: "Texture and Reflection in Computer Generated Images," Commun. of the ACM, pp. 542-546, 1976.
5) G. Miller and R. Hoffman: "Illumination and Reflection Maps: Simulated Objects in Simulated and Real Environments," Proc. SIGGRAPH '84, Course Notes – Advanced Computer Graphics Animation, 1984.
6) B. Cabral, M. Olano and P. Nemec: "Reflection Space Image Based Rendering," Proc. SIGGRAPH '99, pp. 165-170, 1999.
7) W. Heidrich and H. P. Seidel: "Realistic, Hardware-accelerated Shading and Lighting," Proc. SIGGRAPH '99, pp. 171-178, 1999.
8) J. Kautz, P. Vázquez, W. Heidrich and H. P. Seidel: "A Unified Approach to Prefiltered Environment Maps," Proc. Euro Graphics Rendering Workshop '00, pp. 185-196, 2000.
9) B. T. Phong: "Illumination for Computer Generated Pictures," Commun. of the ACM, 18, 6, pp. 311-317, 1975.
10) 池田 聖，佐藤 智和，横矢 直和: "全方位型マルチカメラシステムによるパノラマ動画像の生成"，信学技報，PRMU2002-154, 2002.

奥村　文洋（おくむら　ぶんよう）2001年，名古屋工業大学工学部卒業．2003年，奈良先端科学技術大学院大学情報科学研究科博士前期課程修了．コンピュータビジョン，コンピュータグラフィックスに関する研究に従事．現在，(株)デンソーに勤務．

町田　貴史（まちだ　たかし）1998年，大阪大学基礎工学部卒業．2000年，奈良先端科学技術大学院大学情報科学研究科博士前期課程修了．2002年，奈良先端科学技術大学院大学情報科学研究科博士後期課程退学．コンピュータビジョンに関する研究に従事．現在，大阪大学サイバーメディアセンター助手．

横矢　直和（よこや　なおかず）1974年，大阪大学基礎工学部卒業．1979年，同大学院博士後期課程了．同年，電子技術総合研究所入所．以来，画像処理ソフトウェア，画像データベース，コンピュータビジョンの研究に従事．1986〜1987年，マッギル大学知能機械研究センター客員教授．1992年，奈良先端科学技術大学院大学・情報科学センター教授．現在，同大情報科学研究科教授．正会員．