反射特性を考慮した実時間グレア描画手法
A Real-Time Glare Rendering Technique Taking into Account Surface Reflection Attributes

東京大学大学院，日本 SGI 株式会社	正会員	柿本正憲
日本 SGI 株式会社		向井亨光
日本 SGI 株式会社		芳賀剛士
東京大学大学院	正会員	西田友是
東京大学大学院		苗村　健
東京大学大学院		原島　博

〈**あらまし**〉 本論文では，太陽光のような強い光が反射した場合に生じるグレアをリアルタイムで表現する手法を提案する．現在 OpenGL において使われている Phong の鏡面反射モデルでは，ハイライトの輝度が最大値を超えると，鏡面反射指数（shininess）が大きい物体でも明るさが変わらず，ハイライトの拡がりが小さくなるぶん逆に暗く見えるという欠点がある．本提案では，この反射モデルに修正を加え，ハイライト本来の強い明るさの計算を行なう．この計算によって得た高輝度の反射は，ディスプレイでは物理的に表現不可能である．これに対して，輝度を考慮した新しいグレア表現モデルを考案し，強い反射光の擬似的な描画を行なう．その結果，鏡面反射指数の値にふさわしいハイライトを表現できるようになった．また，OpenGL を用いて本手法を実装し，グレア形状の柔軟な制御とリアルタイム描画を実現した．

キーワード：グレア，反射，環境マッピング，ハイライト，鏡面反射指数

〈**Abstract**〉 This paper proposes a novel glare generating method to render reflected light due to a strong light source on the surface of high-shininess objects. Phong's reflection model, which OpenGL adopts, is physically inexact when the brightness of the center of highlight exceeds maximum intensity. Higher reflection exponent (shininess) value, which should result in sharper highlight, only causes smaller highlight with the brightness unchanged. This makes the highlight look even darker. To solve this issue, we present a new glare generation model with a light energy preserving reflection model. By calculating more physically-correct high intensity reflection than Phong's model, the glare model produces realistic images which more plausibly represent the shininess of each object. An implementation of the glare model is done with OpenGL, which enables real-time rendering and flexible control of the glare shape.

Key words: glare, specular reflection, environment mapping, highlight, shininess

1. はじめに

CAD やゲームや Web3D などの分野では，3D モデルをリアルに，かつインタラクティブな速度で表示することが必須となっている．

リアリティを高めるための研究は，物体の反射モデルを中心にこれまで盛んに行なわれている．一方，最近の HDR（High Dynamic Range）画像の研究により，ディスプレイの物理的な最大輝度を超えるような非常に明るい光を扱うことの有効性が高まってきた．

CG で明るい光を表現する一手段として，グレア効果がある．グレアは，光が人間の眼，特に睫毛や虹彩で回折を起こすために生じる現象で，CG の分野ではまぶしさを表現するための特殊効果として，ゲームや映像制作で用いられている．また，他の産業では，次のような応用例が考えられる．車の CAD モデルの形状評価では，屋外に置いた車に反射する太陽光は，デザイン評価の際の尺度となる．Web3D の通信販売では，多数の照明がきらびやかに反射する宝石や宝飾品を，魅力的に画面に提示することが重要である．

筆者らは，このような強い光に対して，その強さに応じてグレアの形状を計算によって決定し，リアルタイムで描画する手法を提案する．

提案手法は，車のヘッドランプや木漏れ日のような直接光によるグレアを扱うこともできるが，本論文では，鏡面反射物体に映りこむ反射光によって生じるグレアを中心に扱う．確立したシミュレーション技術である鏡面反射モデルと組み合わせることにより，提案手法の有効性を示す．

提案するグレア表現モデルの構成要素としては，OpenGL で使われている Phong の反射モデルを改良して高輝度画素も計算できるようにした鏡面反射モデルを用いた．それによって得られる高輝度の画素値からグレア形状を決定する方式を考案した．基本的な反射の表現は環境マッピング [1] を用いている．マルチパスレンダリングを行なうことにより強い光の反射位置を検出し，そこにおける鏡面反射パラメータも取得し，それらの結果を反映したグレアの生成を行なう．

グレアの生成処理に関しては，人間の眼の構造を考慮した中前らの方法 [2] を採用し，各画素単位でのグレア形状を計算するためにフィルタの代わりに OpenGL の描画プログラムでの実装を行ない，リアルタイム処理を実現した．

2. 関連研究

太陽光に代表される強い光源を表現するための研究は，多く行なわれている．Chiu ら [3] は，CRT と見た目を一致させるために画像の明るさを部分的にスケーリングした．Neumann ら [4] は金属の表面に生じるハイライトをより明るく見せるための工夫を行なった．また，屋外の太陽光の効果を表現するさまざまな手法が提案されている [5][6][7][8][9]．

上記のような手法に対して，人間の目やカメラレンズの特性により生じるぎらつき感，すなわちグレアを描画することで，強い光のリアリティを表現する手法がある（レンズにフィルタをかけて意図的に生じさせる放射状の光の線などの効果は，レンズフレアと呼ばれる）．

従来から，グレアあるいはレンズフレアの表現に関しては，いくつかの研究成果が報告されている．新谷ら [10] は，写真のクロススクリーンフィルタをシミュレートしたレンズフレアを描画する手法を実現した．中前ら [2] は，新谷らの手法を応用して，車のヘッドライトによって目に生じるグレアを表現した．Spencer ら [11] は，グレアのメカニズムを分析してより一般的なグレア表現を行なった．Debevec ら [12] は HDR（High Dynamic Range）画像を用いて太陽光の明るさを表現する際に，同様の手法で高輝度部分にグレアを付加している．

これらの手法は，いずれも空間フィルタによる画像処理で実現されている．この技術はグレアフィルタと呼ばれて，現在広く応用されている．

また，Rokita[13] は，描画結果の中にある高輝度の光源の位置を検出し，そこにグレア画像を拡大／縮小して上描きする手法を提案した．これは空間フィルタと類似した簡易手法であるが，直接光のみが対象で反射光を扱うことができない．

一方，グレア形状を OpenGL で描画する手法は，Kilgard により実装された方法 [14] があるが，予め用意されたグレアのパーツの画像をテクスチャとして重ね合わせながら色を付加したもので，ユーザのデザインの余地が多く，シミュレーションの要素はない．

本論文では，反射モデルを考慮してグレアの形状をシミュレートする新しいグレア表現モデルを提案し，高輝度反射物体をよりリアルに表現する．また，提案するグレア表現の実装には，フィルタを使用せず OpenGL の描画機能を用いた．これにより，グラフィックス・ハードウェアを利用した高速な鏡面反射パラメータ収集を実現するとともに，実行時の柔軟なグレア形状生成を可能にした．また，インタラクティブな操作に対応できる表示速度を実現した．

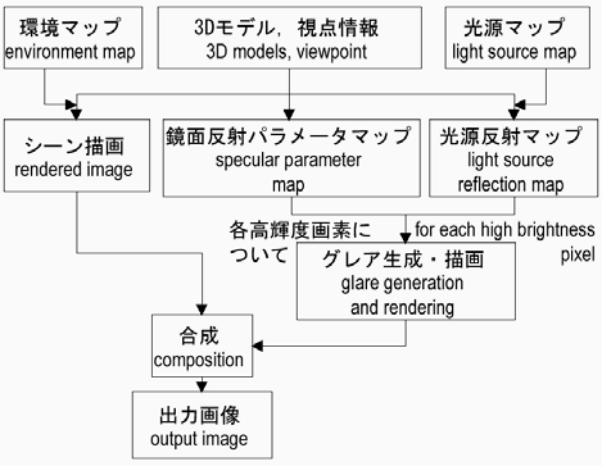

図 1　提案手法の処理の流れ.

Fig. 1　Processing flow of the proposed method.

3.　反射モデルに基づいたグレアの描画

3.1　処理の概要

筆者らの提案する手法は，Phong の反射モデルにおける物体の鏡面反射指数を考慮してハイライトの強さを決定し，その強さにもとづいてグレアの形状を生成するものである．

強い光が反射してグレアを生じるような物体は鏡面反射成分が多いため，映り込みが生じる．リアルタイムの映り込みを表現する一般的な手法は環境マッピングであり，本手法でも環境マッピングの使用を前提とする．

提案手法の処理の流れを図 1 に示す.

通常の環境マップのほかに，HDR 画像などから得られた，強い光源（太陽など）の位置が特定されるテクスチャ（光源マップ）を用意する．まず，光源マップを映り込ませたシーンを描画する．結果の画像には，光源の強さ（色）で反射領域が描かれる（光源反射マップ）.

次に, 物体の鏡面反射係数（specular）と鏡面反射指数（shininess）との値を画素値として持つような画像を描画する（鏡面反射パラメータマップ）.

結果として，これら二つのマップ画像には，各画素ごとにグレアを生じさせるために必要な情報が保存されることになる.

それらの画像をフレームメモリから読み込み，グレアの生じるべき画素を検出し，そこに当たった光の強さと鏡面反射係数, 鏡面反射指数の値からグレアの形状を決定する．その情報をもとに実際に OpenGL 命令によってグレアを描画して，通常の環境マップで別途描画しておいた結果に上書きする.

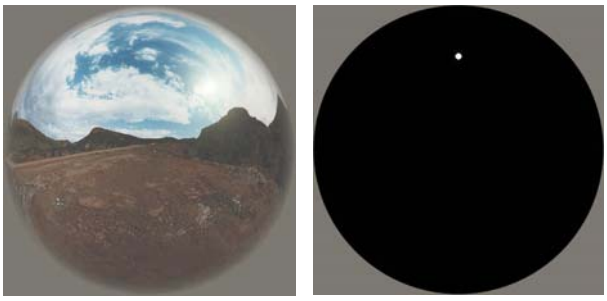

図 2　環境マップ（左）と対応する光源マップ（右）.

Fig. 2　An environment map (left) and corresponding light source map (right).

以下，本章では，処理手順の詳細を説明する.

3.2　光源マップの生成と光源反射マップの描画

映り込みを表現するのに必要な環境マッピングのテクスチャ画像（以下環境マップと呼ぶ）のほか，強い光源の位置と形を設定する別の環境マップを用意する．これを**光源マップ**（light source map）と呼ぶことにする.

光源マップは，環境マップと全く同じシーンを映した画像だが，非常に明るい部分だけが描かれて他の部分は暗くなるように，ダイナミックレンジを高輝度領域に合わせた画像となっている．結果的に，強い光源の場所にその光源の色が描かれ，他の部分はすべて黒になっているような環境マップとなる.

光源マップの生成法は，基本的には通常の環境マップと同様である．一般に，レイトレーシングを用いて環境マップを生成する場合は，球体に周囲のシーンを映りこませる．光源マップを生成するには，周囲のシーンとして太陽のような光源だけを配置し，球体に映りこませればよい.

完全反射の球体を実際にカメラで撮影することにより環境マップを作成する場合は，カメラの露出を絞ってシャッタスピードも速くし，太陽などの映りこんだ部分以外はまっ黒になるような画像を取得する.

このようにして生成した光源マップを環境テクスチャとして描画すると，強い光源が物体に映りこんだ画素は光源の色に，そのほかの場所は黒に塗りつぶされた画像が得られる．この画像を**光源反射マップ**（light reflection map）と呼ぶことにする.

通常の環境マップ及び対応する光源マップの例を，図 2 に示す．図 2 の光源マップとしては，露出を絞った環境マップ画像の代わりに，太陽の位置を中央上部に設定した画像を，別途ペイントソフトによって用意した．また，図 3 には，通常の環境マップでの描画結果と光源反射マップを示す.

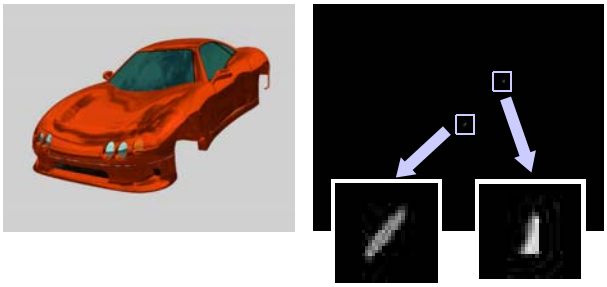

図 3　シーン描画結果（左）と対応する光源反射マップ（右）.
Fig. 3　A rendering result (left) and corresponding light reflection map (right).

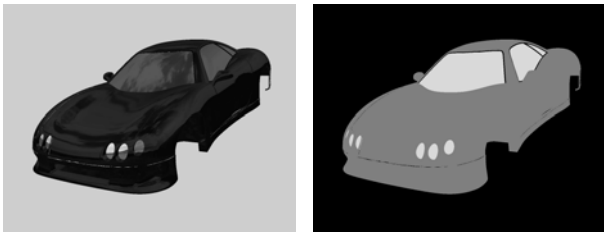

図 4　描画結果（左：図 3 左と同一）と対応する鏡面反射パラメータマップ（右）.
Fig. 4　A rendering result (left) and corresponding specular reflection parameter map (right).

3.3　鏡面反射パラメータマップ

OpenGL では，物体表面属性として，ambient, diffuse, specular, emissive の四種類の成分がある．グレアを生じさせる成分は，このうち specular 成分（鏡面反射成分）である．鏡面反射成分は，物体表面にハイライトと呼ばれる明るいスポットを生じさせる反射成分である．ハイライトの強さと拡がり具合は，物体固有の鏡面反射係数 k_s の値と，鏡面反射指数 n の値とによって決まる．これらを鏡面反射パラメータと呼ぶことにする．

提案方式では，OpenGL の光源モデルで使用される鏡面反射パラメータの値を，グレアの形状を決めるための入力情報として使用する．OpenGL の鏡面反射成分の反射モデルと提案方式の反射モデルの違いは，3.4.2 で説明する．ここでは，物体ごとに与えられた鏡面反射パラメータの値を取得してグレア描画処理に渡す方法について説明する．

グレア描画は，シーン描画結果の各画素に対して行なうから，鏡面反射パラメータも，各画素ごとに与える．それには，同じアングルでシーンを別途描画するが，光源を無視し，物体の色として，その物体の鏡面反射パラメータを RGBA にセットして描画する．この描画結果を**鏡面反射パラメータマップ**（specular reflection parameter map）と呼ぶことにする．

図 4 には，鏡面反射パラメータマップの例を示す．3D 形状モデルは，図 3 で用いたのと同じものを使用している．

3.4　鏡面反射パラメータに応じたグレアの生成

提案手法では，強い光が映りこむ物体の表面属性，特に鏡面反射指数 n によって，グレアの見え方を変える．例えば，車のボディに映りこむ太陽光によるグレアよりも，フロントガラスでのグレアの方が鋭く大きい．これは，ボディの n よりもフロントグラスの n の方が大きいためである．n の値が大きければグレアは鋭く，小さければグレアは鈍くなる．本節では，鏡面反射指数 n の値に応じてグレアの長さと太さを調整する方法をくわしく述べる．

3.4.1　Phong の鏡面反射モデルの問題点

Phong の反射モデル [15] では，物体表面上のある一点 $\boldsymbol{x}$ について，鏡面反射（specular）成分 I_s の式を

$$I_s = k_s I_l (\boldsymbol{R} \cdot \boldsymbol{V})^n = k_s I_l \cos^n \theta \tag{1}$$

によって与えた．ここで，k_s は物体の鏡面反射成分の割合を与える定数（鏡面反射係数）で，0 から 1 の間の値をとる．I_l は光源の強さ，$\boldsymbol{R}$ は，光源が点 $\boldsymbol{x}$ で完全鏡面反射する向きの単位ベクトル，$\boldsymbol{V}$ は点 $\boldsymbol{x}$ から視点に向かう単位ベクトルである．指数 n は鏡面反射指数で，物体の材質によってきまり，値を大きくするほどハイライトの拡がりが小さくなる．θ は，$\boldsymbol{R}$ と $\boldsymbol{V}$ との間の角度である．

一方，Blinn は，Phong の反射モデルの代替案として，次のような反射モデルを提案した [16].

$$I_s = k_s I_l (\boldsymbol{N} \cdot \boldsymbol{H})^n = k_s I_l \cos^n \alpha \tag{2}$$

ここで，$\boldsymbol{N}$ は点 $\boldsymbol{x}$ における単位法線ベクトル，$\boldsymbol{H}$ は，点 $\boldsymbol{x}$ から光源に向かうベクトル $\boldsymbol{L}$ と視点に向かうベクトル $\boldsymbol{V}$ とのなす角を二等分する単位ベクトルである．α は，$\boldsymbol{N}$ と $\boldsymbol{H}$ の間の角度である．n は鏡面反射指数だが，同じ n の値でも Phong のモデルとは若干結果が異なる．反射によるグレアが起こるような物体は，ハイライトの拡がりは小さくなるため，$n = 30$ から $n = 100$ 程度に設定される．

図 5 に，Phong の反射モデルと Blinn による代替モデルを示す．OpenGL では，鏡面反射モデルとして Blinn の式を採用している．

ところで，人間の眼には，ハイライトが鋭いほど中心部はより明るく見える．これは，反射成分の光が狭

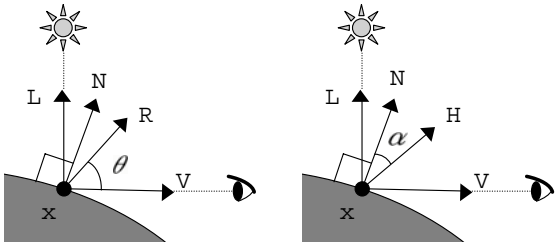

図 5　Phong の鏡面反射モデル（左）と Blinn による代替案（右）.

Fig. 5　Phong's specular reflection model (left) and Blinn's variation (right).

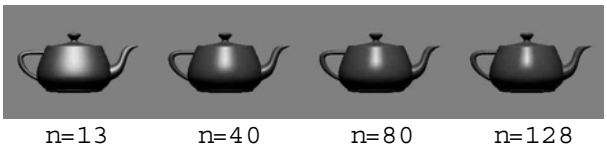

図 6　Phong の鏡面反射モデル（Blinn の式）によるハイライト表現.

Fig. 6　Rendering samples of Phong specular highlight (Blinn's formula).

い範囲内の角度に集約するためであり，本来はこのことを考慮してハイライト中心部の輝度を上げるべきである.

式 (1) または式 (2) で示す反射モデルの問題点は，n の値を大きくしてもハイライトの中心点（$\theta = 0$ または $\alpha = 0$）の明るさは一定（ディスプレイの最大輝度すなわち $I_s = 1$ となる場合が多い）となる点である．結果として，ハイライトの拡がり具合が小さくなるだけで，鋭くはならない．明るさが変わらないため，n が大きくても，ハイライトが小さくなるぶんむしろ暗く見えてしまうという欠点がある.

図 6 に，OpenGL で鏡面反射指数 n を変えたときのモデルの見え方の変化を示す.

3.4.2　高輝度反射表現のための鏡面反射モデル

前記の問題点は，ディスプレイの物理的輝度の上限があることが一つの理由で，本質的な解決は困難である．本手法では，輝度が物理的上限を超えた場合には，相応のグレアを描画して人間の眼にハイライトの鋭さを感じさせることによって，この問題を解決する.

本節では，実際のハイライトの強さがどうなるべきかを論じる．基本的な考え方は，鏡面反射光の総和が鏡面反射指数 n の値に関わらず一定となるように反射光の式を正規化する，というものである．Phong の反射モデルの式 (1) に対しては，田中ら [17] や Lewis ら

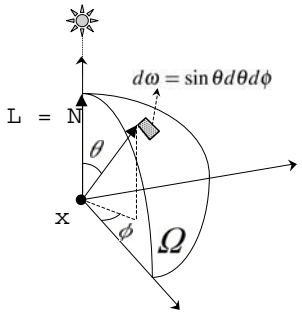

図 7　鏡面反射光の総量の計算.

Fig. 7　Computing the total quantity of reflection light.

[18] が，正規化の式を提案している.

一方，提案方式では，実時間処理を実現するため OpenGL を使用しており，OpenGL は，鏡面反射に関しては Phong のモデルではなく Blinn のモデルを採用している．そこで，Blinn のモデルに対して正規化を行なう．まず，式 (2) に基づいて，鏡面反射指数 n に対する反射光の総量 $T_s(n)$ を計算してみる．計算を簡単化するために，田中らや Lewis らの方法と同様に，光源方向と反射点での法線ベクトルとが等しい（$\boldsymbol{L} = \boldsymbol{N}$）と仮定する.

図 7 のように，反射点 $\boldsymbol{x}$ を囲む，法線ベクトル側の単位半球面 Ω 上の視点 (θ, ϕ) を考えると，前述の仮定により $\alpha = \theta/2$ となる．また，Ω 上の微小面 $d\omega$ を考えると，$d\omega = \sin\theta d\theta d\phi$ となる.

そこで，式 (2) の反射光を Ω 上で積分すると，次のようになる.

$$\begin{aligned} T_s(n) &= \int_\Omega I_s d\omega = k_s I_l \int_0^{2\pi}\int_0^{\frac{\pi}{2}} \cos^n\frac{\theta}{2}\sin\theta d\theta d\phi \\ &= k_s I_l \frac{8\pi}{n+2}\left\{1 - \left(\frac{\sqrt{2}}{2}\right)^{n+2}\right\} \\ &\approx k_s I_l \frac{8\pi}{n+2} \qquad (n \gg 1 \text{ の場合}) \end{aligned} \tag{3}$$

すなわち，反射光の総量は一定ではなく，ほぼ $n+2$ に反比例する．実際，図 6 を見ても，n が大きくなるとハイライトの全体量は明らかに減っている．これでは鏡面反射指数が高いほど反射光の総量が減ることになり，現実的ではない.

本提案では，反射光の総量が n に関わらず一定だと仮定する．そしてそれを反映するために，Blinn の式 (2) を正規化し，次のような式を得た.

$$I_s = k_s I_l \frac{n+2}{8\pi}\cos^n\alpha \tag{4}$$

結果として，ハイライトの鋭さに応じて輝度が高く

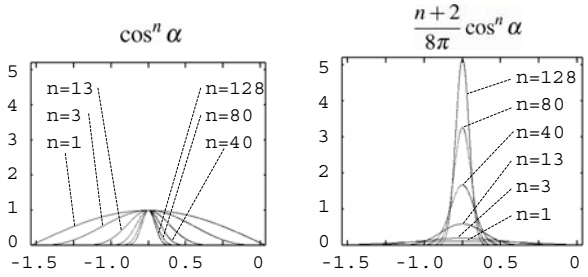

図 8　従来の Specular 反射式（左）と反射光総量を保存する Specular 反射式（右）．それぞれ，$n = 1, 3, 13, 40, 80, 128$ についてプロットした．

Fig. 8　Conventional specular function (left) and the light energy preserving function (right). $n = 1, 3, 13, 40, 80, 128$

なるようなモデルを実現した．図 8 に，$k_s = 1, I_l = 1$ とした場合の，式 (2) および式 (4) のグラフを示す．

式 (4) を単位半球面 Ω 上で積分すると，

$$T_s(n) = \int_\Omega I_s d\omega = k_s I_l \left\{ 1 - \left(\frac{\sqrt{2}}{2} \right)^{n+2} \right\} \tag{5}$$

となり，$n \gg 1$ のときは総反射量は一定となる．

3.4.3　鏡面反射指数とグレア形状の関係

本節では，鏡面反射指数 n とグレア形状，特に睫毛によって生じる streak の長さ L_s 及び太さ W_s との関係を考える．方針は次のとおりとする．

1. streak の長さ L_s は反射点の輝度 I_s に比例させる
2. streak の面積 S は n に無関係で $k_s I_l$ に比例させる
3. streak の長さが短くなっても一対（向きが互いに逆の 2 本）の形状が正方形になったらそれ以上短くしない

以上のような方針で，高輝度領域のある一画素に対して放射状の streak を生じさせる．本節では，同じ方向で向きが 180 度異なる一対（2 本を同じ長さとする）の streak（睫毛 1 本の回折光）だけについて考える．

図 9 に一対の streak の形状を二組示す．L_s は streak 一本分の長さである．W_s は中心での太さでハイライトの拡がりに相当する．S は streak 一本の面積で，常に

$$S = L_s W_s / 2 \tag{6}$$

が成立する．ここで，方針 1 より，比例定数 k_{length} を導入し，

$$L_s = k_{length} I_s = k_{length} k_s I_l \frac{n+2}{8\pi} \tag{7}$$

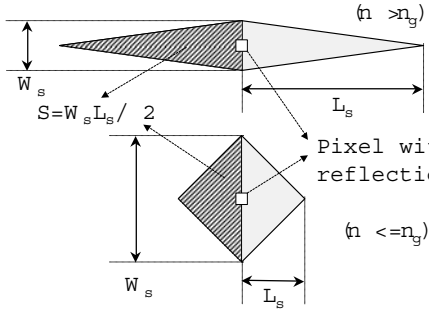

図 9　n の値の異なる二組の streak の形状．

Fig. 9　Two pairs of streaks with different n values.

を得る．次に，方針 2 より，比例定数 k_{size} を導入し，次の式を得る．

$$S = k_{size} k_s I_l \tag{8}$$

さらに，方針 3 より，式 (7) は，n の値がある特定の値 n_g 以上のときにのみ成り立つ．$n = n_g$ のときは，一対のグレア形状が正方形になるときで，$L_s = W_s/2$ となるから，式 (6) と組み合わせると，

$$L_s = \sqrt{S} \tag{9}$$

となる．上記は，$n < n_g$ のときも成立する．以上をまとめると，streak の形状は次のようにして得られる．

$$L_s = \begin{cases} k_{length} k_s I_l \dfrac{n+2}{8\pi} & n > n_g \text{ のとき} \\ \sqrt{S} & n \le n_g \text{ のとき} \end{cases} \tag{10}$$

$$W_s = \frac{2S}{L_s} \tag{11}$$

ここで，n_g の値は，式 (7) および式 (9) より，次のように決まる．

$$n_g = \frac{8\pi\sqrt{S}}{k_{length} k_s I_l} - 2 \tag{12}$$

したがって，比例定数 k_{length} と k_{size} を与えれば，物体の鏡面反射パラメータ k_s, n の値および光源の I_l の値と組み合わせて，式 (10) および式 (11) によってそれぞれ L_s, W_s を計算できる．

streak の長さ L_s と面積 S は，単位が画素であることに注意する必要がある．出力ウィンドウのサイズを変えたときに，3D 形状モデルの大きさと同様にグレアの大きさも変えたい場合は，比例定数 k_{length} をウィンドウの長さに，k_{size} をウィンドウの面積に比例させる必要がある．

定数 k_{length} と k_{size} は，ユーザによる設定が可能だが，シーン内のすべての物体に対して同じ値を使う．

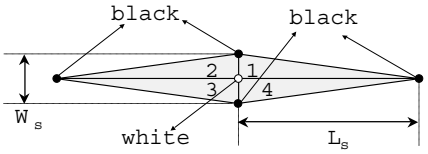

図 10　一対の streak の構造.

Fig. 10　Structure of a pair of diagonal streaks.

例えば，提案手法を 3D のゲーム制作で用いる場合には，制作時に一度設定したらそのコンテンツでは一貫して同じ値を使うべきである．しかし，そのゲームを，画面解像度の違う別のプラットフォームに移植する場合には，画面サイズに比例させて k_{length}，k_{size} の値も変える必要がある．

3.5　光源反射マップに基づくグレアの描画

3.2 の方法で求めた光源反射マップには，光源が完全鏡面反射している部分に明るい画素が生じる．それら明るい画素の各点をそれぞれハイライトの中心点と考え，各画素について，3.4.3 で求めた L_s，W_s に基づいて streak を描画する．

3.5.1　放射状の streak の傾き角と長さ分布の決定

放射状に描かれる数本から 10 数本の streak の傾き角は，初期設定時に，0 度から 180 度の間でランダムに発生させる．描画時には，streak 中心位置のスクリーン上での x 座標の値に応じて，すべての streak の傾きを少しずつずらしていく．これによって，グレアの位置が移動するときに，放射状のグレアが微妙に回転するという効果が得られる．

傾き角と長さとの関係については，睫毛の傾き分布に依存して決まる，という中前らのモデル [2] に従って，水平から 20 度の向きが確率的に最も長くなるように，値が $[0, 1]$ で変化する分布テーブルを用意する．分布テーブルはどの反射物体でも共通だが，基準になる長さは，物体ごとに式 (10) に従って変わる．

3.5.2　streak の描画方法

放射状の streak のうちの，同じ方向で向きが 180 度異なる一対（2 本）の streak（睫毛 1 本の回折光）は，OpenGL の triangle fan の描画機能を用いて描く．

具体的には，図 10 に示すように，4 つの三角形（1〜4）からなる triangle fan で細長いひし形を構成し，一対の streak とする．中心点は白に，周囲の頂点は黒に設定し，各三角形内部は明るさを補間する．各 streak の長さは，式 (10) で求めた Ls の値に，3.5.1 で述べたような，傾き角に応じた分布テーブルの値を乗算して求める．幅 Ws は式 (11) によって求める．

最終的には，数本から 10 数本の streak を放射状に描くが，そのときに，各 streak は半透明物体として描く．こうすることによって，streak が重なり合った分だけ明るくなり，スクリーン上の光量は，streak の重なり具合にかかわらず一定となる．すなわち，反射光の総量が一定であるという仮定を満たすことになる．ただし，現実には，特に streak の面積を決める比例定数 k_{size} の値を大きめに設定すると，streak の面積 S が大きくなり，グレアの中心付近では重なりが多くなりすぎて，輝度の合計が最大値を超えてしまい，それ以上は明るくならない．

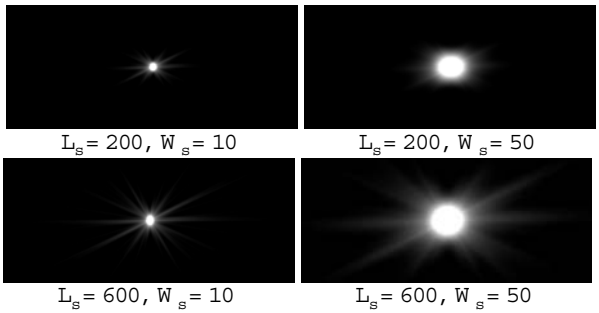

図 11　グレア描画の例.

Fig. 11　Sample rendering of streaks of glare.

以上述べた方法で描いた streak の例を，図 11 に示す．

あらかじめ作成したグレア画像を上描きする Rokita[13] の手法では，高輝度領域が視点に近づくと，グレア画像を拡大／縮小して上描きしている．これに対し提案手法では，グレアの大きさは, 高輝度領域各画素の輝度計算結果（式 (10) および式 (11)）によって決定する．各画素ごとに描くグレアは半透明とするので，高輝度領域が視点に近づいて大きくなった場合は，グレアがより多く重なり合って明るさが加算される．結果的に，グレアは明るく表示される．Rokita の手法がグレア画像の大小で擬似的に高輝度領域の拡がりを表現するに過ぎないのに対し，提案手法は，一般的に使われているグレアフィルタと同様に，重ね合わせによって高輝度領域の拡がりを表現している．また，Rokita の手法では streak は虹彩で生じるとしているが，提案手法では，streak は睫毛で生じるという中前ら [2] のモデルを採用している．

4.　実験結果と応用例

本章では，提案手法をいくつかのモデルに対して適用した例を示す．本論文で示す例は，すべてマウスを使ってリアルタイムで動かしているときの 1 シーンである．

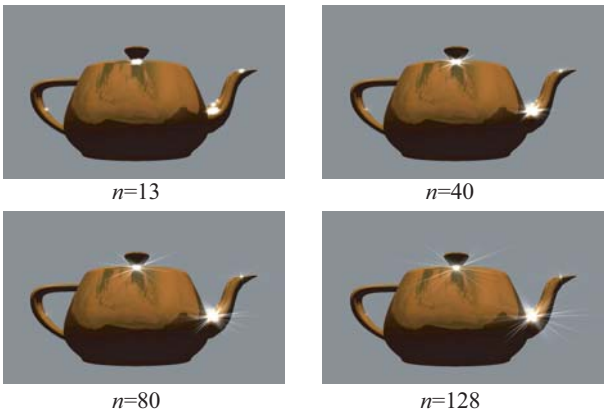

図 12　鏡面反射指数とグレア形状の関係.

Fig. 12　Shininess values and corresponding glare shapes.

図 13　車のデザインレビューへの応用.

Fig. 13　An application to design review of automobile exterior.

4.1　鏡面反射指数とグレア形状

3 章で述べた手法に従って，鏡面反射指数 n を変えながらハイライトとグレアを描画した結果を，図 12 に示す．n の値が大きいときにグレアの streak は長く細くなり，従来の OpenGL のハイライト（図 6）に比べると，反射がより鋭く表現されている．

4.2　車のデザインレビュー

一般に，車の外観をデザインする際には，ボディへの映りこみが重要な評価尺度となる．車は屋外で使うものであるから，CG を使ったディジタルモックアップでのデザイン評価において，太陽光が映りこんで生じるグレアを表現することは特に有用である．

図 13 では，ボディと窓ガラスに映りこむ太陽のグレアを表示している．鏡面反射指数は，ボディよりも窓ガラスの方が大きいため，窓ガラスでのグレアの方が鋭く表現されている．

また，ドライブシミュレータへの応用を考えると，前方を走る車からの太陽光の照り返しも，本手法で表現できる．

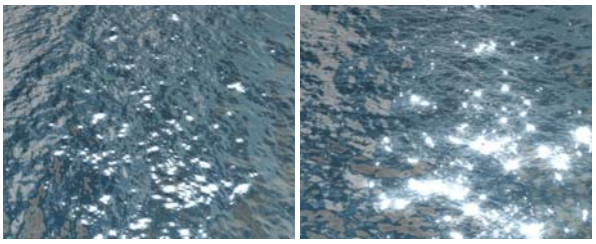

図 14　晴れた日の水面の表現.

Fig. 14　Rendering of water on a shiny day.

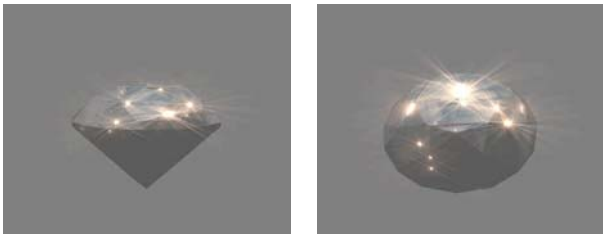

図 15　白熱灯を多数使ったダイヤモンドの展示.

Fig. 15　A diamond with a number of incandescent lamps.

4.3　水面の表現

海や湖などの美しさを表現するのに，水面への太陽光の反射は効果的である．本手法によって水面に生じるグレアを表示した例を図 14 に示す．ここで用いた海面の形状データは，Tessendorf の方法 [19] を用いて生成した．

このような水面のグレア表示は，船舶や航空機のシミュレータにも応用することができる．

4.4　宝石の陳列

宝石を美しく見せるためには，鋭く強い光源を多数用意し，宝石の表面に多くの streak を発生させて輝きを演出する．図 15 には，白熱灯の色の光源を多数配置した光源マップを用いてグレアの描画を行なった例を示す．

4.5　描画処理時間

表 1 に，1 フレームの描画処理時間の測定結果を示す．使用ハードウェアは，SGI Onyx3000（CPU 400MHz, IR4 グラフィックス）である．

この測定では，物体をマウスで操作し，1000 フレーム分について各フレームの処理時間をすべて記録した．表中,「マップ描画」は, 図 1 における「鏡面反射パラメータマップ」と「光源反射マップ」の描画処理に相当する．実装上は，シーン中の各物体の鏡面反射係数 k_s および光源の強さ I_l を一定値にすることによって，マップ描画は 1 回で済むようにした．具体的には，テクスチャマッピングの乗算機能を使って,「鏡面反射パ

表 1　描画処理時間の測定結果
Table 1　Results of rendering time

シーン	ポリゴン数	シーン描画	マップ描画＋読み出し	グレア描画	合 計
図 12	4032	1	8	6-101 (7)	15-110 (16)
図 13	204773	44	51	6-27 (7)	101-122 (102)
図 14	130050	34	40	6-169 (52)	80-243 (126)
図 15	316	0.4	8	6-77 (6)	14-85 (14)

ポリゴン数以外は単位 msec.
カッコ内の数字は中央値.

ラメータマップ」と「光源反射マップ」の画素ごとの積を一つのマップとして描画した．表 1 中の「読み出し」とは，そのマップ画像をフレームメモリから主メモリに転送する処理である．

「シーン描画」と「マップ描画＋読み出し」は，ほぼ安定した値が得られたので，表 1 には平均値を記載した．グレアの描画は，高輝度の画素の数に比例して時間がかかるため，シーン中の反射光の量によって計算時間が大幅に変動する．そのため，表 1 の「グレア描画」には，その最小値・最大値・中央値を記載した．最大値だけ極端に大きいのは，まれにグレアの数が極端に増える場合（比較的フラットな面に光源が映りこむ場合）があって，そこで急に表示が遅くなるためである．図 14 のシーンに対応する 1000 フレーム分のマウス操作時は，大部分のときにグレアが多かったため，中央値も大きい．

この測定結果により，処理全体のスループットとして，まれに 4fps（frame per second）程度に落ちるが，おおむね 7fps〜60fps のフレームレートが得られることがわかる．このように，提案手法によって，インタラクティブな操作を可能にするグレアの実時間描画を実現した．

5.　まとめ

本論文では，物体表面に映りこんだ強い光によって生じるグレアをシミュレートするモデルを提案した．Phong の鏡面反射モデルを修正し，高輝度のハイライトをより正確に計算する手法を採用し，その明るさにもとづいてグレアの形状を決定する．グレア形状の柔軟な制御を行なうために，提案手法を OpenGL によって実装した．グラフィックスハードウェアの描画機能を利用することにより，画素ごとの鏡面反射パラメータを高速に計算し，物体の反射特性を反映した良好なグレア描画結果を得るとともに，インタラクティブな操作を可能にする高速描画を実現した．

今後は，この手法をベースに，色収差によるグレアの分光や，シーン全体の明るさが眼の感度に与える影響などを考慮した，より正確なグレア表現モデルを確立していきたい．また，鏡面反射指数の値に応じて環境マップの映りこみ画像に変化を与えるようなリアリズムの実現や，環境マップを用いずに OpenGL の光源データをもとにグレアを描画する，より汎用の手法の実現を目指す予定である．

謝辞

海面の形状データを提供してくださった，東京大学西田研究室の岩崎慶さんに感謝いたします．

参考文献

[1] J. F. Blinn, M. E. Newell : ”Texture and reflection in computer generated images,” Communications of the ACM, Vol. 19, No. 10, pp. 542–547, 1976.

[2] E. Nakamae, K. Kaneda, T. Okamoto, T. Nishita : ”A lighting model aiming at drive simulators,” Proc. of ACM SIGGRAPH 1990, Vol. 24, pp. 395–404, August 1990.

[3] K. Chiu, M. Herf, P. Shirley, S. Swamy, C. Wang, K. Zimmerman : ”Spatially nonuniform scaling functions for high contrast images,” Proc. of Graphics Interface ’93, pp. 245–253, 1993.

[4] L. Neumann, A. Neumann, L. Szirmay-Kalos : ”Compact metallic reflectance models,” Computer Graphics Forum (Eurographics ’99), Vol. 18, pp. 161–172, 1999.

[5] Y. Dobashi, K. Kaneda, T. Nakashima, H. Yamashita, T. Nishita, K. Tadamura : ”Skylight for interior lighting design,” Computer Graphics Forum (Eurographics ’94), Vol. 13, pp. 85–96, 1994.

[6] H. W. Jensen, F. Durand, M. Stark, S. Premoze, J. Dorsey, P. Shirley : ”A physically-based nightsky model,” Proc. of ACM SIGGRAPH 2001, pp. 399–408, August 2001.

[7] E. Nakamae, T. Ishizaki, T. Nishita, S. Takita : ”Composition 3D images with anti-aliasing and various shading effects,” IEEE CG&A, Vol. 9, No. 2, pp. 21–29, 3 1989.

[8] T. Nishita, E. Nakamae : ”Method of displaying optical effects within water using accumulation-buffer,” Proc. of ACM SIGGRAPH 1994, pp. 373–380, August 1994.

[9] A. J. Preetham, P. Shirley, B. Smits : ”A practical analytic model for daylight,” Proc. of ACM SIGGRAPH 1999, pp. 91–100, August 1999.

[10] M. Shinya, T. Saito, T. Takahashi : ”Rendering techniques for transparent objects,” Proc. of Graphics Interface ’89, pp. 173–182, 1989.

[11] G. Spencer, P. Shirley, K. Zimmerman, D. P. Greenberg : ”Physically-based glare effects for digital images,” Proc. of ACM SIGGRAPH 1995, pp. 325–334, August 1995.

[12] P. E. Debevec, J. Malik : ”Recovering high dynamic range radiance maps from photographs,” Proc. of ACM SIGGRAPH 1997, pp. 369–378, August 1997.

[13] P. Rokita : ”A model for rendering high intensity lights,” Computers & Graphics, Vol. 17, No. 4, pp. 431–437, 1993.

[14] M. J. Kilgard : ”Fast OpenGL-rendering of lens flares,” http://www.opengl.org/developers/code/mjktips/lensflare/.

[15] B. T. Phong : ”Illumination for computer generated pictures,” Communications of the ACM, Vol. 18, No. 6, pp. 311–317, June 1975.

[16] J. F. Blinn : ”Models of light reflection for computer synthesized pictures,” Computer Graphics (Proc. of SIGGRAPH 1977), Vol. 11, No. 2, pp. 192–198, July 1977.

[17] T. Tanaka, T. Takahashi : ”Shading with area light sources,” Computer Graphics Forum (Eurographics ’91), Vol. 10, pp. 259–268, 1991.

[18] R. R. Lewis : ”Making shaders more physically plausible,” Fourth Eurographics Workshop on Rendering, pp. 47–62, 1993.

[19] J. Tessendorf : ”Simulating ocean water,” Simulating Nature: Realistic and Interactive Techniques, SIGGRAPH 2001 Course Note #47, 2001.