Vol.2016-CG-164 No.4
2016/9/21

「Material Editing in Complex Scenes by Surface Light Field Manipulation and Reflectance Optimization」の実装

谷田川 達也[1,a)]

概要：本稿では，2013 年に Eurographics で Nguyen らが発表した論文である「Material Editing in Complex Scenes by Surface Light Field Manipulation and Reflectance Optimization」の実装と，その発展に向けた展望について述べる．Nguyen らの論文は Phong の陰影モデルを仮定した実時間レンダリング用のシーンに関して，ユーザから与えられたペイントストロークに基づき双方向反射分布関数 (Bidirectional Reflectance Distribution Function, BRDF) を編集する手法である．この手法の特徴は，シーンをレンダリングする際に間接反射光も考慮している点で，それに伴いシーンの編集は直接光を編集するタイプと間接光を編集するタイプの二種類が提案されている．本稿では Nguyen らの手法の一部の実装した結果を紹介するほか，手法をより一般化し，半透明物体を扱うための試みについても言及する．

1. はじめに

コンピュータ・グラフィクスで，より実写に近い画像を生成するため，三次元的なシーンからの画像を生成するレンダリング技術は幅広い研究がなされている．ここでの三次元的なシーンは，シーン中の物体形状を表現するための三次元メッシュ，およびそれぞれの物体の質感を表現するための多くのパラメータを含む．一般に物体の見た目をユーザの意図通りに変化させるためには，これらの質感パラメータを何度も調整せねばならず，その調整を容易にするためにレンダリング・シーンの編集技術も数多くの研究がなされている．これらの編集法の詳細については，いくつかのサーベイ論文を参考にしていただきたい [6].

本稿で紹介する Nguyen らの「Material Editing in Complex Scenes by Surface Light Field Manipulation and Reflectance Optimization」[5] は，このような編集技術の中でも，シーンを構成する物体の材質感を編集する技術に分類される．Nguyen らの手法は，特にシーン中の光の反射が Phong の陰影モデルに従うと仮定し，個々の物体表面に割り当てられている双方向反射分布関数 (Bidirectional Reflectance Distribution Function, BRDF) を編集する．Nguyen らの手法で扱われる BRDF $f_r(\mathbf{x}, \omega_i, \omega_o)$ は物体表面上の各点 $\mathbf{x}$ ごとの定義されており，光の入射方向 ω_i と出射方向 ω_o を変数とする関数である．ここで点 $\mathbf{x}$ における法線を $n(\mathbf{x})$, 光沢度を $\alpha(\mathbf{x})$，拡散反射係数を $K_d(\mathbf{x})$，鏡面反射係数を $K_s(\mathbf{x})$ とすると，Phong の BRDF は次の式で表される．

$$f_r(\mathbf{x}, \omega_i, \omega_o) = K_d(\mathbf{x}) \max(0, \omega_i \cdot n(\mathbf{x})) + K_s(\mathbf{x}) \max(0, \omega_o \cdot (\omega_i - 2n(\mathbf{x})(\omega_i \cdot n(\mathbf{x})))^{\alpha(\mathbf{x})}$$

Nguyen らの手法では，視点の位置と光源の位置を固定した状態で上記の BRDF を編集する．このとき，直接反射光のみを考えたとすれば変更可能なパラメータは K_d および K_s のみとなる．従って，Nguyen らの手法では，ユーザが入力したペイントストロークを基にこれらの K_d および K_s を編集することを目指す．

2. 実装の詳細

以下では，Nguyen らの手法を実装するにあたり，彼らの用いているシーンのレンダリング手法ならびに，シーンの編集方法について紹介する．

2.1 直接光・間接光のレンダリング

Nguyen らの手法では，直接光と間接光の両方を表現できるレンダリング手法を用いている．今回の実装では，直接光については，一般的な Blinn-Phong の陰影モデルとシャドウマップ法を用いた．間接光については，Preconvolved Radiance Caching [7] という手法 (以下，PRC 法と表記する) が用いられている．この手法は実時間レンダリングに

[1] 早稲田大学
, 3-4-1 Ohkubo, Shinjuku, Tokyo 169-8555, Japan
[a)] t.yatagawa@kurenai.waseda.jp

ⓒ 2016 Information Processing Society of Japan

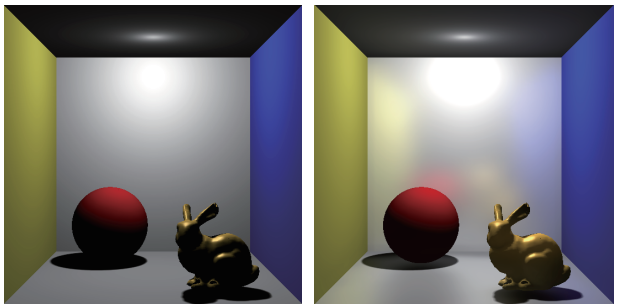


図 1 直接光のみのレンダリングと間接光を付与したレンダリング

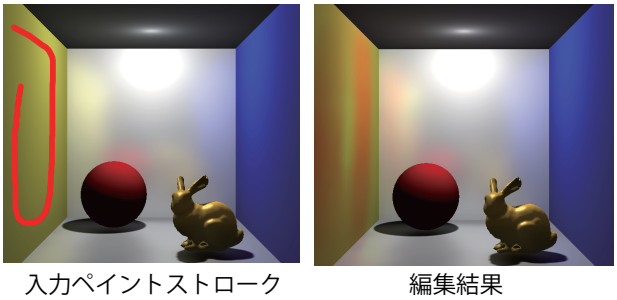


図 2 入力のペイントストロークと対応する編集結果.

おいて間接光を計算する手法の一つである Radiance Cache Splatting 法 [3] (以下，RCS 法と表記する) で扱いの難しかった光沢面のレンダリング部分を改善した手法である．

PRC 法では，まず，シーン全体から間接反射光を計算するサンプル点を抽出する．今回の実装では，サンプル点の抽出に一般的な Poisson Disk Sampling 法を用いた．抽出されたサンプル点の一つ一つについて，法線方向を見たときの半球上の見えを，放物面投影法によりテクスチャに描画する．ここで描画される半球上の見えの一つ一つは Lumitexel と呼ばれる 32×32 程度の小画像であり，すべてのサンプル点からの見えを一枚のテクスチャに描画することができる．このテクスチャを用いると，視点が決まった時に，反射方向がどのように見えるのかが瞬時に参照可能となる．PRC 法では光沢面による反射を考慮するため，各小画像をぼかした結果を異なるぼかし強度ごとに MIPMap に格納しておく．レンダリング時には物体表面の光沢度に応じて参照する MIPMap のレベルを変化させるだけで，間接反射光成分が得られる．

PRC 法では，間接光は低周波の成分が支配的であると仮定しており，間接反射成分を各サンプル点に対して得たあとは，サンプル点を中心とした RBF 補間でスクリーン上の各画素における間接光を計算する．この RBF 補間をシェーダ上で実現するためにサンプル点を一定の大きさを持つ円として描画し，円の描画領域に含まれるピクセルで RBF 関数の影響を評価し，それをアルファ合成により重ね合わせることで，高速な RBF 補間を実現している．

PRC 法により間接光を計算した結果を図 1 に示す．この図では，コーネルボックスの床および背面にある壁に強い光沢があるが，これらの光沢面による間接反射の成分が適切に表現されていることが分かる．

2.2 直接・間接反射光の編集

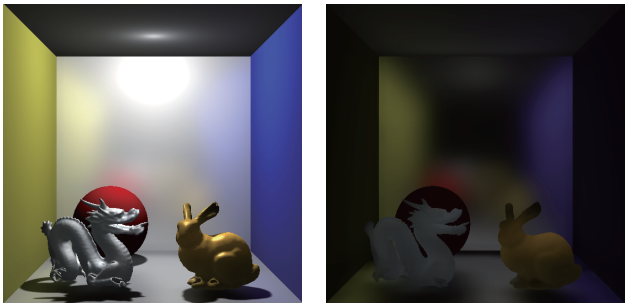
Nguyen らの手法による BRDF の編集は，ペイントストロークに従って行われる．ペイントストロークは RGB の三次元の情報を持っており，ペイントを塗った場所の色をペイントの色に変化させるような編集を意図する．以下では簡単のために直接光の編集法について述べた後，間接光の編集法について言及する．

直接光を編集する場合には，ペイントされた物体表面の BRDF を直接変化させることで目的の編集結果が得られる．例えば，BRDF の拡散反射係数 K_d だけを編集するときには，入射光の強さとペイントされた色から，RGB の各チャネルで独立に編集後の K_d を求めることができる．一方で Nguyen らの扱う BRDF で変更できるパラメータは K_d, K_s ならびに α の合計七種類がある (K_d と K_s は三チャネル分のパラメータと考えた)．これらの変数全てを変化させて，ペイント色に最も近い編集結果を得る問題は，変数の数が七つに対して，制約式の数が RGB の三つ分しかない設定不良問題である．従って，Nguyen らは拡散反射と鏡面反射の成分のどちらか一方のみが変化した際に，もっともペイント色を反映するようなパラメータの組み合わせを編集結果として採用している．なお光沢度の強さ α については，いくつかの候補値をあらかじめ決めて置き，その中で最もペイント色を再現するものを選ぶ．

上記の編集はシーン上から選ばれたサンプル点上で実施され，レンダリング時に，編集前後の BRDF 変化量を補間することで，編集結果を描画する．編集結果を補間する際には，物体表面上の二点間の類似度を，位置の近さ，法線の近さ，BRDF の近さの三つの指標から計算する．以上の方法で直接反射光を編集した結果を図 2 に示した．この図ではシーン左側の黄色い壁を赤色に変化させるようなペイントが指示されており，編集結果ではペイントされた部分の色が変更されていることが分かる．また，ペイントされた壁が背面の壁に映りこんでいる部分に関しても適切に編集後の色が反映されていることが分かる．この映り込みの変化は，ペイント箇所の BRDF 変化を計算した後で Lumitexel を再計算することで実現できる．

Nguyen らの手法では，上記のようにペイントされた部分を直接編集する方法とは別に，ペイント部分に間接的に影響を与える物体表面の BRDF を変化させる方法も提案されている．この方法では，ペイントされた物体表面に影響を与える箇所をペイント部分の BRDF を使って計算する．例えばペイントされた箇所が完全拡散反射表面であったとすれば，ペイント箇所から半球状に見える全ての物体

ⓒ 2016 Information Processing Society of Japan



直接光あり　　直接光なし

図 3　半透明物体の挿入結果.

表面に対して，拡散反射係数を考慮してペイントを伝搬させる．この伝搬のためには，各物体表面からの別の物体表面の可視性を用いる必要があるが，この可視性はシーン読み込み時に前計算しておく．ペイントを伝搬した後は，直接光の編集と同じようにして，サンプル点における BRDF の変化を計算する．

3. 半透明物体のシーンへの挿入

本稿では，Nguyen らの BRDF 編集法に加え，試験的に半透明の物体をシーンへ挿入する実験を実施した．実験における半透明物体のレンダリングには間接光のレンダリング手法と同じく Splatting 技術を用いた Chen らの手法 [1] を採用した．

3.1 半透明物体のレンダリング

Chen らの手法は Translucent Shadow Maps [2] に代表されるような光源からの放射照度を格納したサンプル点を用いる半透明物体の実時間レンダリング手法である．あらかじめ，半透明物体上にサンプル点を配置し，そのサンプル点における放射照度を計算しておく．シーンのレンダリング時には，各サンプル点からの透過光の影響を Splatting と呼ばれる技術を用いて評価する．Chen らの手法では，各サンプル点のスクリーン座標系での位置を中心とした円をシェーダにより描画する．描画する際には，円の描画領域に含まれる画素で，サンプル点からの透過光の影響を評価する．評価された透過光の影響はアルファ合成により積算され，最終的な透過光の影響が求められる．

今回の実装による半透明物体を挿入したシーンのレンダリング結果を図 3 に示す．この図では，シーン左側のドラゴンが半透明になっており，図 3 右に示す通り，直接光の影響を除去すると，ドラゴンに透過光が計算されていることが分かる．

3.2 性能評価

今回の実装は，OpenGL および GLSL を用いて実装し，Intel Core i7 4.0 GHz の CPU ならびに NVIDIA GeForce 960 上でシステムを実行した．その結果，サンプル点を 12000 個程度用いた際に，図 3 の画像をおよそ 6 fps 程度で描画することができた．サンプル点を 60000 個程度まで増やすことで間接反射光，および透過光の計算結果に改善が見られたが，この場合は描画速度が 1 fps 程度まで減少してしまった．サンプル点を増やした際の計算のボトルネックは間接反射光の計算部分であり，半透明部分とそれ以外の部分とでサンプル点の密度を変更するなどして，描画速度の向上が可能であると考えられる．

4. まとめと今後の展望

本稿では，表題の Nguyen らの BRDF 編集法に加え，レンダリングシーンに半透明物体を挿入する実験を実施した．今回の実験では，描画速度が十分でなく，また半透明物体を編集するところまでは至っていない．描画速度の向上に関しては，間接光および透過光の計算に用いるサンプル点の取り方を改善するほか，半透明物体のレンダリングにスクリーン空間上での効率的な半透明物体描画法 [4] を応用するなどの方向性が考えられる．半透明物体の編集に関しては，シーン中に単一の半透明物体がある場合の編集法である SubEdit[8] などを拡張し，物体間の相互反射を考慮した編集手法とすることが求められる．

参考文献

[1] Chen, G., Peers, P., Zhang, J. and Tong, X.: Real-time rendering of deformable heterogeneous translucent objects using multiresolution splatting, *The Visual Computer*, Vol. 28, No. 6, pp. 701–711 (2012).

[2] Dachsbacher, C. and Stamminger, M.: Translucent Shadow Maps, *Proceedings of the 14th Eurographics Workshop on Rendering*, EGRW '03, pp. 197–201 (2003).

[3] Gautron, P., Křivánek, J., Bouatouch, K. and Pattanaik, S.: Radiance Cache Splatting: A GPU-friendly Global Illumination Algorithm, *ACM SIGGRAPH 2005 Sketches*, SIGGRAPH '05, ACM (2005).

[4] Jimenez, J., Zsolnai, K., Jarabo, A., Freude, C., Auzinger, T., Wu, X.-C., der Pahlen, J., Wimmer, M. and Gutierrez, D.: Separable subsurface scattering, *Computer Graphics Forum*, Vol. 34, No. 6, Wiley Online Library, pp. 188–197 (2015).

[5] Nguyen, C. H., Scherzer, D., Ritschel, T. and Seidel, H.-P.: Material Editing in Complex Scenes by Surface Light Field Manipulation and Reflectance Optimization, *Computer Graphics Forum*, Vol. 32, No. 2, Wiley Online Library, pp. 185–194 (2013).

[6] Ritschel, T., Dachsbacher, C., Grosch, T. and Kautz, J.: The State of the Art in Interactive Global Illumination, *Computer Graphics Forum*, Vol. 31, No. 1, pp. 160–188 (2012).

[7] Scherzer, D., Nguyen, C. H., Ritschel, T. and Seidel, H.-P.: Pre-convolved Radiance Caching, *Computer Graphics Forum*, Vol. 31, No. 4, Wiley Online Library, pp. 1391–1397 (2012).

[8] Song, Y., Tong, X., Pellacini, F. and Peers, P.: SubEdit: a representation for editing measured heterogeneous subsurface scattering, *ACM Transactions on Graphics (TOG)*, Vol. 28, No. 3, ACM, p. 31 (2009).

ⓒ 2016 Information Processing Society of Japan