Vol.2016-CG-162 No.2
2016/2/8

Laplacian Editingを用いたセルシェーディングにおける陰影境界の編集手法

川口　龍樹[1,a)]　土橋　宜典[1,b)]　山本　強[1,c)]

概要：昨今，アニメーションの分野において，3DCG をセル画調で表現するセルシェーディングが多く用いられている．これはノンフォトリアルレンダリング（NPR）の一種で，陰影の境界が明瞭であるといった特徴を持つ．NPR では物理的な正確さはあまり求められず，いかにユーザの要求を表現するかが重要であり，レンダリング結果を編集するための技術が必要になる．本稿では，ユーザが編集を希望する陰影境界線の範囲をマウス操作で選択しそれに Laplacian Editing と呼ばれる頂点列の編集手法を用いることで，ユーザのインタラクティブな操作によりセルシェーディングの陰影境界を編集する手法を提案する．

キーワード：NPR，セルシェーディング，Laplacian Editing

Boundary Manipulation for Cel-Shading by Using Laplacian Editing

Kawaguchi Tatsuki[1,a)]　Dobashi Yoshinori[1,b)]　Yamamoto Tsuyoshi[1,c)]

1. はじめに

昨今，アニメーション製作の分野において，動きが複雑であったり大量の動きが必要であったりするシーンなどに対して，手描きの絵に 3 次元コンピューターグラフィックス（3DCG）を合成するような技法が用いられることがある．その際，手描き絵と 3DCG の合成に違和感が生じないように 3DCG をレンダリングをする必要があり，中でもセル画アニメーションに近しい絵の表現のためのレンダリング手法にセルシェーディングがある．セルシェーディングはノンフォトリアリスティックレンダリング（NPR）の一種で非写実的な絵を表現するための手法である．その特徴として，物体の輪郭線が描かれることと，描かれる陰影の明暗がはっきりとしその境界が明瞭であることが挙げられる．

NPR においては絵を作る際の物理的な正確さやそれに伴う複雑な計算は重要ではないことがあり，ユーザが要求する表現をいかに最終的な絵として出力できるかが求められる．特にセルシェーディングにおいては手描き絵との合成時の親和性を高めることが重要である．そのため NPR におけるレンダリングの結果をユーザが直接編集できるような技術が求められ，その為の手法も多く研究されている．

セルシェーディングの編集手法として Todo らの方法 [1] がある．これはペイント操作によって陰影を編集する手法であり，明るくしたい部分や暗くしたい部分，あるいは陰影境界を調節したい部分をユーザ操作によって編集することを可能にする．しかしペイント操作による編集は自由度が高い反面，望みの結果を得るためにはユーザの修練も必要になるほか，手入力による境界部のブレなどが生じやすいといった問題点が存在する．

本研究ではセルシェーディングの編集において特に陰影境界の編集に着目し，より単純かつ簡単にその陰影境界を自然に編集するための手法について様々な方法を検証し考察を行った．編集に際して核となる技法として，Deferred Shading[2] と Laplacian Editing[3] を採用した．Deferred Shading は 3 次元のシーンの法線や深度，カラーといった

[1] 北海道大学大学院情報科学研究科
Graduate School of Information Science and Technology, Hokkaido Uniersity
[a)] kawaguchi@ime.ist.hokudai.ac.jp
[b)] doba@ime.ist.hokudai.ac.jp
[c)] yamamoto@ime.ist.hokudai.ac.jp

情報を一度 2 次元画像として書き出し，それに光源やレンダリングに付随する情報併せて画像として合成することで最終的な絵を得る手法である．Laplacian Editing は，接続関係を持った頂点列の編集手法で，頂点列にいくつかの位置制約を与えると，点列の概形をなるべく保つように点列全体の変形を行う手法である．本稿では，この 2 手法に Todo らの手法で用いられるオフセット関数の考えを加えた陰影境界の編集を実現するための手法について論じる．

以降，本論文の構成は，2 節で従来研究として Todo らの方法，及び本研究の先行研究として行った陰影境界の編集手法について解説を行い，3 節で本稿で用いる技法である Deferred Shading と Laplacian Editing について記述した．4 節で本研究で行った陰影境界の編集手法の提案と実装を解説し，5 節で提案法の比較と考察を記述した．最後に 6 節でまとめとする．

2. 関連研究

本節では，まずユーザのドラッグ＆ドロップによるセルシェーディングの操作手法として Anjyo[4] らの手法を紹介し，次に本研究で利用した Todo らの方法について解説を行う．

2.1 Tweakable Light and Shade for Cartoon Animation

Anjo らの手法はセルシェーディングついて，ユーザのドラッグ＆ドロップによるシーン上のハイライト形状や陰影領域の編集を可能にする．陰影の編集については，点 $\mathbf{p}_a$ から $\mathbf{p}_b$ への移動に対して，光源を元の方向 $\boldsymbol{L}_d$ から新しい方向 $\boldsymbol{L}_d^{new}$ を算出することで実現している．ハイライトの編集では，ハイライトを作る要素であるハーフベクトル $\boldsymbol{H}(\mathbf{p})$ をユーザ操作に基づいて，平行移動や拡大縮小，回転を行うように編集することで実現している．この手法は陰影の編集時に光源方向を変更しているためシーン全体の陰影に影響が及び，ユーザが局所的な陰影の調節を求めるような状況に対応できない．

2.2 Locally Controllable Stylized Shading

Todo らの方法はペイント操作で陰影を局所的に編集する手法であり，オフセット関数というアイディアを用いて編集を実現している．最も単純なセルシェーディングでは，点 $\mathbf{p}$ における法線 $\boldsymbol{N}(\mathbf{p})$ と光源方向ベクトル $\boldsymbol{L}(\mathbf{p})$ について，拡散反射光 $I(\mathbf{p}) = \boldsymbol{N}(\mathbf{p}) \cdot \boldsymbol{L}(\mathbf{p})$ を閾値 t で処理する．閾値判定 $I(\mathbf{p}) \geq t$ について，これが真であればそこは明領域，偽であれば暗領域として結果が出力される．Todo らの提案するオフセット関数はこの閾値処理を拡張する．点 $\mathbf{p}$ におけるオフセット関数値 $o(\mathbf{p})$ を考えたとき，閾値判定はオフセット付き拡散反射光 $S(\mathbf{p}) = I(\mathbf{p}) + o(\mathbf{p})$ を用いて $S(\mathbf{p}) \geq t$ と拡張され，オフセット関数値 $o(\mathbf{p})$ 次第で明暗を切り替えることが可能になる．ユーザのペイント操作により，ペイント領域における拡散反射光とそれまでのオフセット値に基づき新しいオフセット関数を決定し，編集結果が得られる．

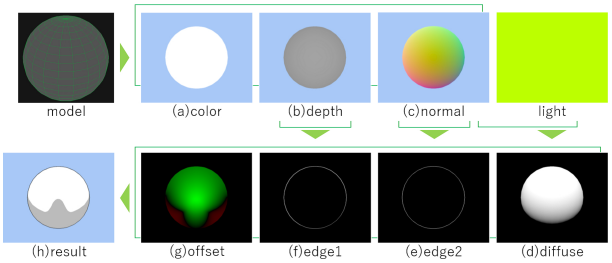

図 1 Deferred Shading によるセルシェーディングの過程

3. Deferred Shading と Laplacian Editing

本節では，陰影境界編集の核とする技法である Deferred Shading と Laplacian Editing について解説を行う．

3.1 Deferred Shading

Deferred Shading は 3 次元のシーンの法線や深度，カラーといった情報を一度 2 次元画像として書き出し，それに光源やレンダリングに付随する情報を併せて画像として合成することで最終的な絵を得る手法である．Todo らの手法ではオフセット関数は 3 次元物体の頂点上に定義されているが，より単純な操作を実現するためこれを 2 次元画像として処理することを考え，本研究ではシーン全体を 2 次元画像として扱うために Deferred Shading を導入した．ここでは，Deferred Shading を用いて**図 1** に示す過程によりセルシェーディングを行う．レンダリングの第一パスでは，カラー画像 $\boldsymbol{C}$（図 1a），深度画像 $\boldsymbol{D}$（図 1b），法線画像 $\boldsymbol{N}$（図 1c）を書き出す．セルシェーディングにおける輪郭線を描画するため法線及び深度の微分画像 $\boldsymbol{N'}, \boldsymbol{D'}$（図 1e,f）を作成し，陰影を描画するため光源方向（ここでは平行光源）$\boldsymbol{L}$ と法線の内積から拡散反射光画像 $I = \boldsymbol{N} \cdot \boldsymbol{L}$（図 1d）を作成する．加えて Todo らの提案するオフセット関数もまた 2 次元画像 O（図 1g）として同様に扱い，閾値 t により $I + O = S \geq t$ と処理し $\boldsymbol{N'}, \boldsymbol{D'}$ と合成して結果画像 R（図 1h）が得られる．

3.2 Laplacian Editing

Laplacian Editing は，接続関係を持った頂点列の編集手法で，頂点列にいくつかの位置制約を与えると，制約を考慮しつつ点列の概形をなるべく保つように点列全体の変形を行う手法であり，本研究では目標陰影境界線の編集のために用いる．頂点列 $\boldsymbol{v}$ についてその接続関係を元にラプラシアン演算 L より微分座標 $\boldsymbol{\delta} = L(\boldsymbol{v})$ を求め，任意の頂点 $\boldsymbol{v}_k$ を位置 $\boldsymbol{u}_k$ に制約した際に，それらの微分座標が可能な

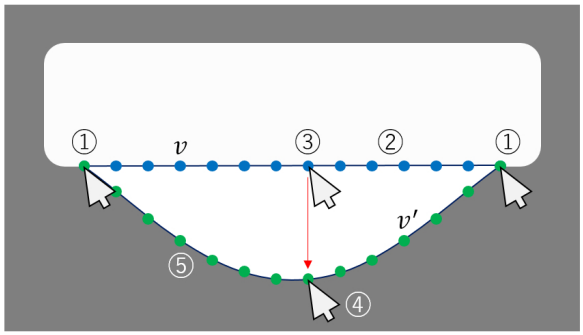

図 2 Laplacian Editing による陰影境界線の編集操作

限り維持されるような編集後の頂点列 $\boldsymbol{v}'$ を求める．これは式 1 の最小化問題に帰着する．

$$\arg\min_{\boldsymbol{v}'} \sum_i |\boldsymbol{\delta}_i - L(\boldsymbol{v}'_i)|^2 + \sum_k |\boldsymbol{u}_k - \boldsymbol{v}'_k|^2 \qquad (1)$$

但し上式では，ラプラシアンの回転が許容されていないため自然な編集結果が得られない．そこで，任意の線形変換 T を導入した式 2 を解く．

$$\arg\min_{\boldsymbol{v}'} \sum_i |\boldsymbol{T_i}\boldsymbol{\delta}_i - L(\boldsymbol{v}'_i)|^2 + \sum_k |\boldsymbol{u}_k - \boldsymbol{v}'_k|^2 \qquad (2)$$

ここでは解法は省略する．

4. 陰影境界編集手法の提案と実装

本節では，セルシェーディングにおける陰影境界の編集を実現する提案手法を解説する．ここでは計 5 つの手法を考案し，それぞれの検証を行った．各手法におけるユーザの共通の操作を**図 2** に示す．まず，ユーザは編集を希望する陰影境界の範囲を 2 点のクリックにより指定する（①）．すると 2 点間に存在する陰影境界上の点列 $\boldsymbol{v}$ が Laplacian Editing の編集対象として抽出される（②）．次に，ユーザは得られた点列 $\boldsymbol{v}$ から 1 点を選択し（③），ドラッグ＆ドロップ操作により陰影境界の移動先を指定する（④）．最初に指定した 2 点と移動先の点を制約点として Laplacian Editing が適用され，編集後の目標となる境界線 $\boldsymbol{v}'$ が得られる（⑤）．最後にこれらの情報を元にして各手法ごとにオフセット関数が算出され，陰影境界が変形された結果の絵が生成される．ここで，画像上の点 $\mathbf{p}$ におけるオフセット関数は編集前を $O_0(\mathbf{p})$，編集後を $O_1(\mathbf{p})$ とし，同様に閾値判定に用いるオフセット付き拡散反射光 S も $S_0(\mathbf{p}) = I(\mathbf{p}) + O_0(\mathbf{p})$，$S_1(\mathbf{p}) = I(\mathbf{p}) + O_1(\mathbf{p})$ と定義する．

4.1 Simple Filling

Laplacian Editing の編集前後の点列の内側を単純に塗りつぶすようにオフセット値を与えることで編集の実現を考え，これを Simple Filling と呼ぶこととする．本手法の概要を**図 3** に示す．ここで，編集前の点列を $\boldsymbol{v}$，編集後の点列を $\boldsymbol{v}'$ とする．画像上の点 $\mathbf{p}$ と $\boldsymbol{v}$ と $\boldsymbol{v}'$ がつくる閉領域の内外判定を行う関数を式 3 に示す．

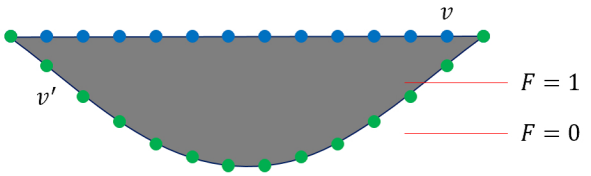

図 3 Simple Filling の概要

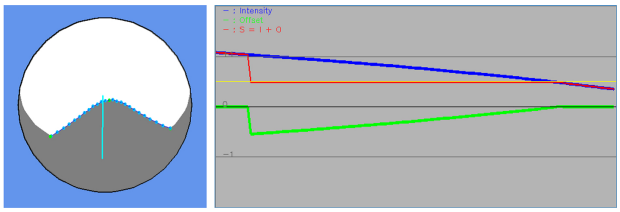

図 4 Simple Filling による編集結果（左図）と左図の水色線上での拡散反射光（青），オフセット（緑），オフセット付き拡散反射光（赤）（右図）

$$F(\mathbf{p}) = \begin{cases} 1 & (\mathbf{p} \text{ is in closed region between } \boldsymbol{v} \textit{ and } \boldsymbol{v}') \\ 0 & (\text{otherwise}) \end{cases} \qquad (3)$$

編集後のオフセット関数は式 4 により求める．

$$O_1(\mathbf{p}) = O_0(\mathbf{p}) + (t - S_0(\mathbf{p}) + \varepsilon \cdot \mathrm{sign}(t - S_{max})) \cdot F(\mathbf{p}) \quad (4)$$

ここで，閾値 t と S_0 の差が閾値の残差を表し，この値をオフセット関数として新たに用いることでその位置における明暗が切り替わる．但し，それだけでは閾値と同値の S 値が得られるため値が明暗どちらかを明瞭にするため，任意の微少値 ε の項を加算する．S_{max} は閉領域内で閾値との残差の絶対値が最大となる S 値でその符号を ε に乗じる．本手法における編集結果と，編集部分における I, O, S の値をグラフに表したものを**図 4** に示す．本手法は，高速に破綻のない結果を得られるという利点がある一方で，不連続な値が得られるため，光源の移動や閾値の変更，多段階の閾値を持つものに対応できないという問題が生じることが確認された．

4.2 Offset Diffusion

Simple Filling は不連続な値を生成するという問題点があった．そこで，連続的なオフセット値を得るために Diffusion Curves[5] という手法を限定的に利用することを考えた．Diffusion Curves はベクトル画像における濃淡の表現方法として提案された手法で，色を設定した曲線を用いて任意形状のグラデーションの作成を可能にする．ここでは，ラスタ画像上で色の代わりにオフセット値を用いて

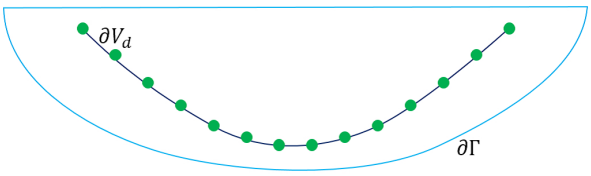

図 5 Offset Diffusion の概要

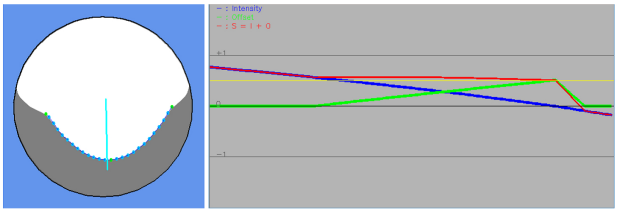

図 6 Offset Diffusion による編集結果（左図）と左図の水色線上での拡散反射光（青），オフセット（緑），オフセット付き拡散反射光（赤）（右図）

Diffusion Curves を適用することで連続的な値の算出を試み，これを Offuset Diffusion と呼ぶこととする．Diffusion Curves において曲線 Ω 上の色源 c により生成される画像 $f(x,y)$ は式 5,6 のラプラス方程式を解くことで求められる．

$$\frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2} = 0 \tag{5}$$

$$f((x,y) \in \Omega) = c \tag{6}$$

本手法では，色源 c としてオフセット関数値を用いて目標境界線及び編集領域周囲に設定し，編集領域内で上式を解くことで新しいオフセット関数を算出する．本手法の概要を**図 5** に示す．ここで，Laplacian Editing により得られた目標境界線 ∂V_d に設定する値を式 7，編集領域周囲 $\partial\Gamma$ に設定する値を式 8 とする．

$$c(\mathbf{p} \in \partial V_d) = t - S_0(\mathbf{p}) \tag{7}$$

$$c(\mathbf{p} \in \partial\Gamma) = O_0(\mathbf{p}) \tag{8}$$

編集後のオフセット関数は式 9 により求める．

$$O_1(\mathbf{p}) = O_0(\mathbf{p}) + f(x,y) \tag{9}$$

本手法における編集結果と，編集部分における I, O, S の値をグラフに表したものを**図 6** に示す．本手法は，連続的な値を得ることが可能であるが，得られた値がすべて閾値を下回る（或いは上回る）保証がなく，陰影に穴が空く問題が生じることが確認された．失敗例を**図 7** に示す．また，編集領域周囲 $\partial\Gamma$ をどのように設定するかという点も問題である．今回は $\boldsymbol{v}$ と $\boldsymbol{v}'$ がつくる閉領域から 20px 離れた範囲までを編集領域とし，その境界を $\partial\Gamma$ としている．

4.3 Intensity Diffusion

Offset Diffusion では編集領域内の陰影の明暗をすべて確実に切り替える保証がなく，編集領域に陰影の穴が空くことがあった．そこで，次に Diffusion Curves の輝度値（S 値）への適用を考え，これを Intensity Diffusion と呼ぶこととする．基本的には Offset Diffusion と処理の過程は同様であるが，色源 c とオフセット関数の算出が異なる．Offset Diffusion における色源の式 7 は，Intensity Diffusion では式 10 に置き換わる．式 8 は共通である．

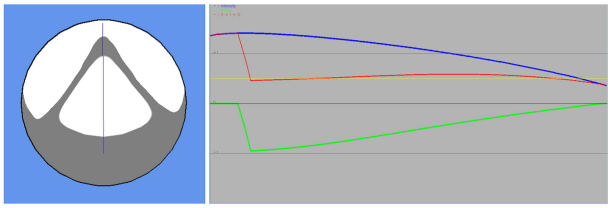

図 7 Offset Diffusion における失敗例（左）とその際の輝度分布

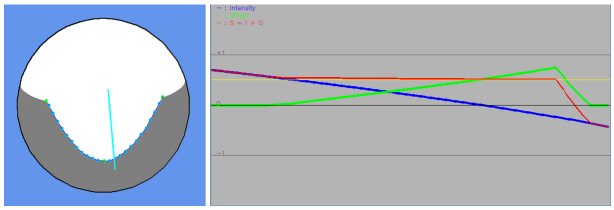

図 8 Intensity Diffusion による編集結果（左図）と左図の水色線上での拡散反射光（青），オフセット（緑），オフセット付き拡散反射光（赤）（右図）

$$c(\mathbf{p} \in \partial V_d) = S_0(\mathbf{p}) \tag{10}$$

編集後のオフセット関数は式 11 により求める．

$$O_1(\mathbf{p}) = O_0(\mathbf{p}) + f(x,y) - S_0(x,y) \tag{11}$$

本手法における編集結果と，編集部分における I, O, S の値をグラフに表したものを**図 8** に示す．本手法は，Offset Diffusion における問題の発生を低減することはできたが，完全な改善とはならず陰影に穴が空く問題が生じることがなお確認された．

4.4 Diffusion with Intensity Gradient

Intensity Diffusion では，Offset Diffusion の問題点に対して僅かな効果がみられたが，完全な改善には至らなかった．そこで，Diffusion Curves のラプラス方程式に輝度勾配に関する項を加えることで改善を図り，これを Diffusion with Intensity Gradient と呼ぶこととする．Diffusion Curves では式 5,6 に示されるラプラス方程式を解くが，本手法では S_0 に関する項を導入し式 12,13 に示されるポアソン方程式に拡張して結果 f を求める．

$$\frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2} = \frac{\partial^2 (-S_0)}{\partial x^2} + \frac{\partial^2 (-S_0)}{\partial y^2} \tag{12}$$

$$f((x,y) \in \Omega) = c \tag{13}$$

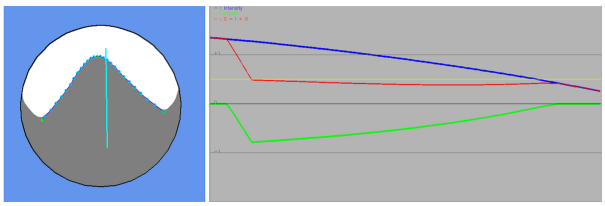

図 9 Diffusion with Intensity Gradient による編集結果（左図）と左図の水色線上での拡散反射光（青），オフセット（緑），オフセット付き拡散反射光（赤）（右図）

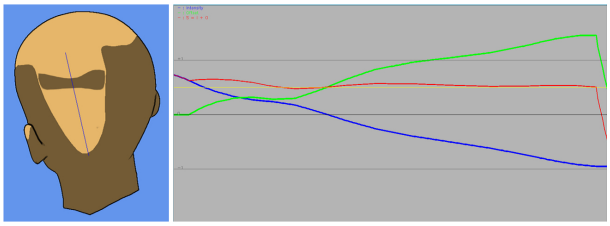

図 10 Diffusion with Intensity Diffusion における失敗例（左）とその際の輝度分布

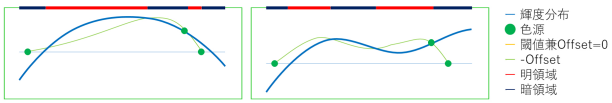

図 11 Diffusion Curves によって陰影の穴が生じる輝度分布の例（左）と本手法によって陰影の穴が生じる輝度分布の例（右）

色源の設定及びオフセット関数の算出は前小節と同様に行う．本手法における編集結果と，編集部分における I, O, S の値をグラフに表したものを**図 9** に示す．本手法は，Offset Diffusion および Intensity Diffusion において生じる陰影に穴が空く問題を防ぐことが可能になった．しかし，異なる形状モデルに適用した際に穴が空く現象が確認された．失敗例を**図 10** に示す．これは，編集領域の輝度分布が原因で，**図 11**（左）に示すように輝度分布の山に対して Diffusion Curves を用いた手法は穴を生じさせるが，本手法では図 11（右）に示すように輝度分布に谷がある際に穴が生じることがあるからであると考えられる．また Diffusion Curves を用いる方法は，計算コストが高いという問題もある．

4.5 Laplacian Mesh Editing

Diffusion with Intensity Gradient でも，編集領域に穴が空く問題の解決には至らなかった．そこで，Diffusion Curves を利用しない方法を検証した．Laplacian Editing は接続関係を持った頂点列に適用することが可能であり，前小節まではこれを線分の編集に適用していたが，本手法ではメッシュ構造について適用することを考えた．編集線を構成する点列とその周囲に楕円状に配置された点群を仮定する．これらの頂点に対して Delaunay 三角形分割 [6] に

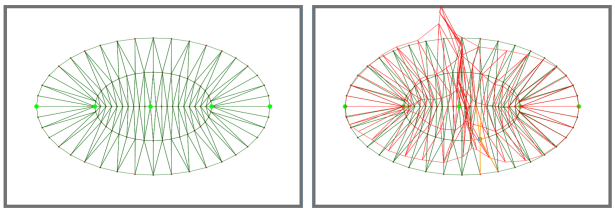

図 12 Delaunay 三角形分割によるメッシュ（左）とその編集例（右）

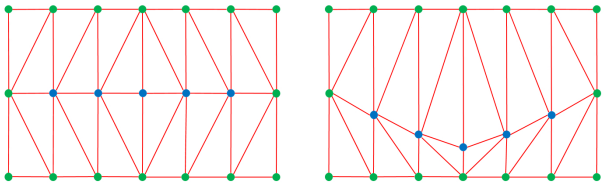

図 13 Triangle Interpolation の概要

よるメッシュ化を行い，メッシュを構成する三角形頂点の編集前後の値を用いてオフセット関数を作成することを考えた．実際に作成したメッシュとその Laplacian Editing による編集例を**図 4.5** に示す．メッシュ構造では陰影境界の編集対象である線分と周囲の点列間の接続関係の強さを調節することができないため，望ましい編集結果が得られないことが確認された．そのため，本手法の陰影編集への適用は行わなかった．

4.6 Triangle Interpolation

メッシュ構造での Laplacian Editing では望みの編集結果を得ることが困難であった．そこで，線分とその周囲に配置した点群を用いた単純な点列の接続関係を用いて目標線の編集を行い，オフセット関数の算出にのみ Delaunay 三角形分割による三角形頂点の値を用いることを考える．これを Triangle Interpolation と呼ぶこととする．本手法の概要を**図 13** に示す．本手法では，ユーザ指定の 2 点から得た線分に加え，閾値 t に対して $\pm dt$ だけずらした位置での線分も同様に取得する．各線分の端点間に接続関係を与え，編集時はこの構造全体に対して Laplacian Editing を適用する．その際，陰影境界線の端点および境界線以外の線分の構成点を固定点とする．オフセット関数の算出では，まず編集前の点群に対して Delaunay 三角形分割を用いてメッシュを構築しその接続関係を記録する．次に編集後の点群に対して，編集前から作成した接続関係を元に，三角形メッシュを新たに作成する．最後に，編集前の三角形 $(\boldsymbol{v}_0, \boldsymbol{v}_1, \boldsymbol{v}_2)$ と編集後の三角形 $(\boldsymbol{v}'_0, \boldsymbol{v}'_1, \boldsymbol{v}'_2)$ について，式 14 を用いて編集後の三角形頂点上に値 f を与え，その内側の値を三角形内の補間により求めて結果画像 f' に保存する．すべての三角形に対して f' を計算し式 15 により編集後のオフセット関数を求める．

$$f(\boldsymbol{v}'_i) = S_0(\boldsymbol{v}_i) \tag{14}$$

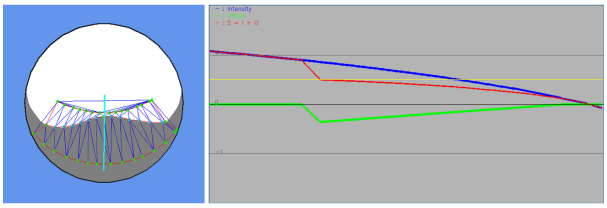

図 14 Triangle Interpolation による編集結果（左図）と左図の水色線上での拡散反射光（青），オフセット（緑），オフセット付き拡散反射光（赤）（右図）

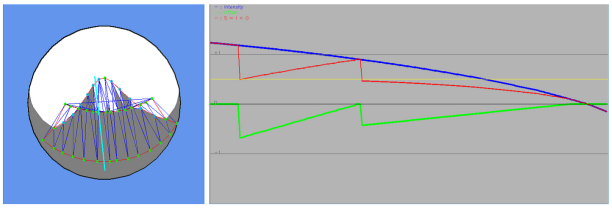

図 15 Triangle Interpolation における失敗例（左）とその際の輝度分布

$$O_1(\mathbf{p}) = O_0(\mathbf{p}) + f'(\mathbf{p}) \tag{15}$$

本手法における編集結果と，編集部分における I, O, S の値をグラフに表したものを**図 14** に示す．本手法は，高速に連続的な値を得ることができるという利点がある．一方で，ユーザが 2 点を選択した時点で編集可能な領域が制限されてしまい，それを超えた編集により三角メッシュが大きく破綻すると正しい結果が得られない問題が確認された．破綻してしまう例を**図 15** に示す．

5. 実験結果の比較と考察

前節では Laplacian Editing を用いた陰影境界の編集手法について理論の提案と実装により明らかになった問題点を記述した．本節では，各手法の比較及び考察を述べる．各手法において算出されるオフセット関数の値についての比較を表 1 に示す．Simple Filling 以外の手法では連続

表 1 生成されるオフセット関数値についての比較

手法	オフセット関数値
Simple Filling	不連続
Offset Diffuse	連続（非線形）
Intensity Diffuse	連続（非線形）
with Intensity Gradient	連続（非線形）
Triangle Interpolation	連続（線形）

なオフセット関数を得ることができる．今回用いたセルシェーディングにおいては結果画像は明暗で二値化されるため，オフセット関数値の不連続性は目に見える結果に影響を与えていない．しかし，セルシェーディングの閾値を変更したり，シーンの光源の位置が変更されたりする際，オフセット関数が不連続であると結果が破綻することが分かっている．また，セルシェーディングには陰影が多段階

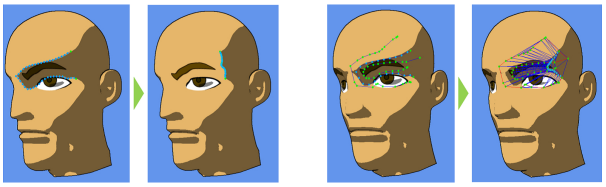

図 16 大きな曲率における成功例（左：Simple Filling）と失敗例（右：Triangle Interpolation）

に設定されるものや，境界にグラデーションが使用されるものもあり，そのようなシェーディングにも対応ができない．但し，今回提案した手法は Simple Filling 以外の連続な値を生成するものであっても，編集領域を制限して連続な値を生成しているため上記の不連続な値に対して問題が生じる例について良好な結果が得られないことがある．

各手法のオフセット関数の算出にかかる処理時間についての比較を表 2 に示す．ここでの処理時間は，ユーザのドラッグ＆ドロップについてドロップしてから計算結果が表示に反映されるまでにかかる時間である．但し，編集する領域の広さによって速度は変動するため一例としての値である．本研究では，CPU は Intel®Core™i7-5820K，GPU

表 2 速度に関する比較

手法	処理速度 [ms]
Simple Filling	82
Offset Diffuse	4233
Intensity Diffuse	4756
with Intensity Gradient	4541
Triangle Interpolation	42

は NVIDIA®GeForce GTX750Ti をそれぞれ用いている．各手法の処理時間と Laplacian Editing の処理速度を合わせて Simple Filling と Triangle Interpolation は 10 – 20fps 程度の速度で結果を反映することができ，リアルタイム処理性能も確認された．

Triangle Interpolation は連続な値をリアルタイムに算出できる点で他に優位な手法である．しかし，Simple Filling は編集領域が制限されず，Diffusion Curves を用いた手法は操作後に計算の対象となる編集領域が決定されるのに対し，Triangle Interpolation は，ユーザが編集範囲を決定した時点で編集領域が決定される．そのため，編集領域を超えた操作では正しい結果を得ることができない．Diffusion Curves を用いる手法は，速度も遅く領域内に陰影の穴を生じてしまう問題を抱えているが，生成される輝度分布は非線形で滑らかに変化するため，陰影境界が最も綺麗に描かれ，閾値変更や光源移動に対して自然な結果が得られる．また，大きな曲率を持つ陰影境界を対象にした際，**図 16** に示されるように Simple Filling 以外の手法では正しく編集できない．

6. おわりに

本研究では，Deferred Shading に Todo らのオフセット関数を加えて，そこに Laplacian Editing を用いたセルシェーディングの陰影境界の編集手法を提案し検証を行った．Laplacian Editing は陰影境界の目標線を作成するために用いて，その目標線に沿うようなオフセット関数の算出方法 5 つ試みた．手法ごとに利点及び欠点があることが確認され，場合に応じて最も適した手法を選択することが求められる．本研究の今後の進展として，さらに新しい陰影境界の編集手法の考案が求められる．特に，ユーザへの操作要求は単純でありながら，編集の自由度はより高く結果に破綻を生じない手法が必要である．

参考文献

[1] Todo, H., Anjyo, K., Baxter, W. and Igarashi, T.: Locally Controllable Stylized Shading, *ACM Transactions on Graphics - Proceedings of ACM SIGGRAPH 2007*, Vol. 26, No. 3, p. Article No.17 (2007).
[2] Deering, M., Winner, S., Schediwy, B., Duffy, C. and Hunt, N.: The Triangle Processor and Normal Vector Shader, *ACM SIGGRAPH Computer Graphics*, Vol. 22, No. 4, pp. 21–30 (1988).
[3] Sorkine, O., Lipman, D., Alexa, M., Rössl, C. and Samarin, H.-P.: Laplacian Surface Editing, *Eurographics/ACM SIGGRAPH symposium on Geometry processing*, pp. 175–184 (2004).
[4] Anjyo, K., Wemler, S. and Baxter, W.: Tweakable Light and Shade for Cartoon Animation, *Proceedings of the 4th international symposium on Non-photorealistic animation and rendering*, pp. 133–139 (2006).
[5] Orzan, A., Bousseau, A., Winnemöller, H., Barla, P., Thollot, J. and Salesin, D.: Diffusion Curves: A Vector Representation for Smooth-Shaded Images, *Communications of the ACM*, Vol. 56, No. 7, pp. 101–108 (2013).
[6] Lee, D. T. and Schachter, B. J.: Two algorithms for constructing a Delaunay triangulation, *International Journal of Computer & Information Sciences*, Vol. 9, No. 3, pp. 219–242 (1980).