仮想彫刻に基づくリアルタイム法線マップ生成システム

脇田 航† 井門 俊†

近年，少ないポリゴン数で，メモリや処理時間を節約しつつ，擬似的に精細な 3D オブジェクトを表現可能な法線マッピングが注目されている．法線マッピングを行うには，法線ベクトル (XYZ) が色 (RGB) として格納された法線マップをもとに，フラグメント単位でレンダリングを行う必要がある．しかし，法線マップを人間が直接描くことは困難である．そこで我々は，仮想彫刻に基づくリアルタイム法線マップ生成システムの提案を行う．本システムでは，3D オブジェクトの表面を直接削る・盛ることにより法線マップの生成を行う．これにより，法線マップを意識することなく，リアルタイムかつインタラクティブに法線マップを作成することが可能となった．

A Real-time Normal Map Generation System Based on Virtual Sculpting

Wataru Wakita† and Shun Ido†

Recently, normal mapping which can pseudoexpress finespun 3D object with low polygons is now attracting attention. However, it is difficult for humans to draw directly because normal vector (XYZ) is stored in a normal map as color (RGB). Therefore, we propose a real-time normal map generation system based on virtual sculpting. This system generates a normal map by scraping or plumping directly the surface of 3D object. As a result, generating a normal map in real-time and interactively without considering a normal map became possible.

1. はじめに

近年，3DCG 技術および GPU（Graphics Processing Unit）の発達にともない，低コストで高速かつリアルな 3D レンダリングが可能となってきた．また，リアルタイム 3DCG 技術として，プログラミング可能な GPU の登場および DirectX や，ATI，NVIDIA などによる OpenGL 拡張にともない，より高速かつ柔軟な頂点処理とフラグメント処理が可能となった[1)~3)]．Xbox や PlayStation などのコンシューマ機にもプログラミング可能な GPU が搭載されるようになり，プログラマブルシェーダによるレンダリングはさらに身近なものとなりつつある．

通常，精細かつリアルな 3D オブジェクトを構築するには，多くのポリゴンやテクスチャなどが必要であるが，モデリングおよびレンダリングにおいて多大なコストがかかってしまう．そのため，レンダリングコストを抑える手法として，ジオメトリやテクスチャのサイズを変化させる LOD（Level of Detail）手法に関する研究や，擬似的に精細な 3D オブジェクトを表現する手法に関する研究がさかんに行われている．Blinn[4)] は，高さマップから法線ベクトルを算出し，ジオメトリの法線を変化させることで，ライティングにおいて擬似的な凹凸を表現するバンプマッピングを提案した．応用例として，尾上ら[5)] は，LOD とバンプマッピングの手法を用いて，少ないポリゴン数で，擬似的に砂丘の表面上に風紋を表現するシステムを開発した．一方，Cook[6)] は，高さマップから法線方向にジオメトリを変化させることで，凹凸を表現するディスプレイスメントマッピングを提案した．この手法では実際にジオメトリを変化させるため，バンプマッピングに比べてコストがかかる．近年，フラグメント単位でシェーディングを行うことで，より精細な 3D オブジェクトを擬似表現可能なレンダリング手法がいくつか提案されている．Kaneko ら[7)] は，フラグメント単位で高さマップから視差分だけテクスチャ座標をずらすことで，擬似的な凹凸の表現を行うパララクスマッピングを提案した．この手法では，視差を考慮することで，より立体感のある凹凸表現が可能である．一方，Oliveira ら[8)] は，フラグメント単位で高さマップに対してレイキャスティングを行い，テクスチャ座標をずらすことで，擬似的な凹凸の表現を行うレリーフマッピングを提案した．この手法はパララクスマッ

† 愛媛大学大学院理工学研究科
Graduate School of Science and Engineering, Ehime University

ピングよりも正確な処理を行うが，コストがかかる．これらのフラグメント単位のマッピング手法に加え，法線マッピングを併用することで，より精細な擬似凹凸表現が可能である．法線マッピングは，法線ベクトル (XYZ) が色 (RGB) として格納された法線マップをもとに，フラグメント単位でジオメトリの法線ベクトルを，ワールドスペースやタンジェントスペースなどで変化させる手法である[3]．法線マッピング単体でもより精細な擬似凹凸表現が可能なため，近年注目されている．応用例として，Wrotek ら[9),10)] は，衝突シミュレーションにおけるジオメトリどうしの衝突の際に，法線マッピングによる凹み表現を行っている．Yamazaki ら[11)] は，法線マップの圧縮アルゴリズムの開発を行っている．法線マップは 2 次元のテクスチャであるが，人間が直接描いて作成することは非常に困難である．そのため，法線マップの生成手法として，高さマップをフィルタにかける手法[9),10)] や，ハイポリゴンモデルとローポリゴンモデルの法線ベクトルの差分から生成する手法[12),13)] などが用いられている．高さマップをフィルタにかけて作成するツールはいくつか存在するが，これらはあらかじめユーザが高さマップを用意しなければならず，目的の法線マップを得るまでに手間がかかってしまう．また，ハイポリゴンモデルとローポリゴンモデルから生成するツールもいくつか存在するが，これらもユーザが目的の法線マップを生成するまでに数ステップ必要であり，リアルタイムかつインタラクティブに生成することはできない．たとえば，Xbox や PlayStation などのような，ハードウェアの制限を受ける環境でのモデリングにおいて，実機ではローポリゴンモデルを用い，品質向上のために，モデリングの段階でハイポリゴンモデルを用意し，ローポリゴンモデルの UV マップにハイポリゴンモデルの法線ベクトルを焼き付ける方法がしばしば用いられている．この方法では，目的の法線マップを得るため，何度もハイポリゴンモデルを修正し，ローポリゴンモデルに焼き付ける作業を繰り返す必要があり，モデリングに多大なコストがかかる．また，この方法により得られた法線マップは，最終的にローポリゴンモデルとともに法線マッピングで擬似的な凹凸としてレンダリングされる．このため，ローポリゴンモデルを初期形状とし，ハイポリゴンモデルのような精細な法線マップをリアルタイムかつインタラクティブに生成することができれば，低コストで擬似精細なモデリングが可能になると考えられる．

そこで本研究では，仮想彫刻に基づき，リアルタイムに法線マップを生成可能なシステムの開発を行う．具体的には，3D オブジェクトの表面を直接削る・盛ることにより，インタラクティブな操作で直感的に法線マップの生成を行う．本システムを用いることにより，ユーザは法線マップを意識することなく，リアルタイムかつインタラクティブに法線マップを生成可能なことを確認する．

2. システム設計

2.1 システム概要

本研究では，インタラクティブな操作でリアルタイムに法線マップを生成することを目的とする．我々は，法線マッピングが最終的に擬似的な凹凸としてレンダリングされることに注目し，仮想彫刻に基づき，3D オブジェクトを直接削る・盛ることにより，リアルタイムに法線マップを生成可能なシステムの開発を行う．また，法線マップ生成だけでなく，3D オブジェクトやテクスチャ編集なども考慮したシステムとして開発を進めていく．

2.2 仮想彫刻に基づく法線マップ生成

これまでに，様々な仮想彫刻に関する研究が行われている．Mizuno ら[14)] は，ペンタブレットの筆圧を用いて，CSG 形状表現された 3D オブジェクトに対して，インタラクティブに切削を行う彫刻・木版画システムの開発を行っている．Cristiano ら[15)] は，パーティクルベースの 3D オブジェクトに対し，リアルタイムに切削を行うシステムの開発を行っている．Jagnow ら[16)] は，仮想彫刻に基づき，力覚提示装置を用いて，3D オブジェクトの表面に対し，リアルタイムにディスプレイスメントマッピングを行うシステムの開発を行っている．これらはすべて 3 次元のジオメトリを書き換えることで仮想彫刻を行っているが，本システムでは，ユーザがドラッグした部分に対応する法線マップを動的に変化させ，フラグメント単位で法線マッピングを行うことにより，擬似的な凹凸を表現するため，2 次元のテクスチャを書き換えることで仮想彫刻を行う．そのため，物理ベースの仮想彫刻ではなく，UV ペイントのような，お絵描き感覚で 3D オブジェクトを削る・盛ることにより，リアルタイムに擬似的な凹凸表現を行う．

2.3 形状表現方法

3D オブジェクトの形状表現方法には，サーフェスモデル，CSG 表現されたソリッドモデル，点群や濃度で 3D オブジェクトを表現するパーティクルやメタボールなどがある．本システムでは，一般的なモデリングソフトで現在最も広く普及しているサーフェスモデルを扱う．また，光の反射係数やテクスチャなどが

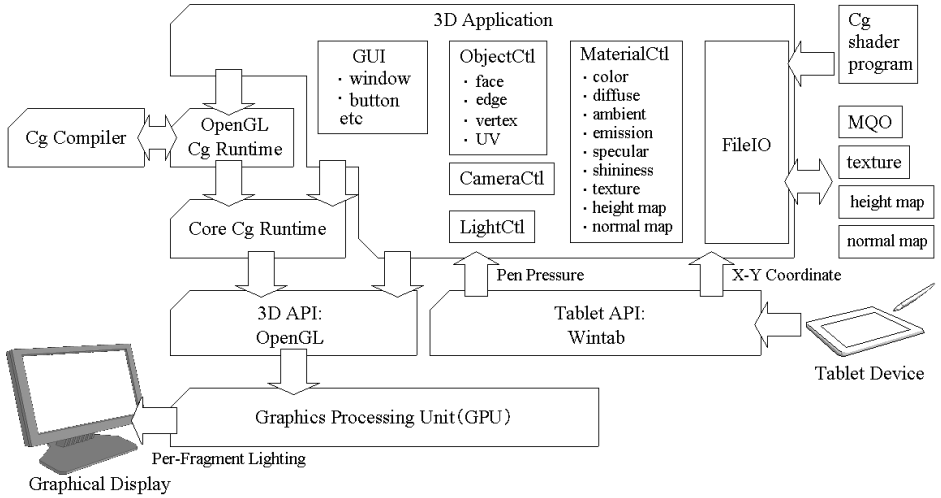


図 1 システム構成
Fig. 1 System architecture.

設定されたマテリアルを 3D オブジェクトの表面に割り当て，フラグメント単位でライティングを行う．

2.4 入力および編集方法

本システムでは，お絵描き感覚で仮想彫刻を行い，3D オブジェクトの詳細を作り込んでいく．2 次元画像から 3D シーンを作り込む手法として，IBMR（Image Based Modeling and Rendering）の分野では，2 次元平面に対して 2 次元画像から深さマップを編集し，階層化や合成を行うことで 2 次元画像から 3 次元シーンを生成する手法[17),18)] がしばしば用いられる．また，法線マップを用いた例として，2 次元平面に対してタブレットの姿勢から直接法線マップを編集し，2 次元画像に方向付けを行うことで，3 次元的なライティングを行えるもの[19)] や，正射影でレンダリングされた 3D オブジェクトから作成した 2 次元画像を法線マップとし，マーキング操作で 3 次元形状を生成する手法[20)] がある．しかし，本システムのように，3D オブジェクトを直接扱う場合，これら 2 次元平面に対して編集を行う方法を用いると，ユーザは UV 展開図に対して操作を行うことになり，彫刻操作時にやや直感性や利便性に欠ける．そのため，本システムでは，UV ペイントのように，UV 展開された 3D オブジェクトの表面に対して直接彫刻を行う方法をとり，ユーザがドラッグした部分に対応するテクスチャ座標周りに高さペイントを行うことで，オブジェクト表面に高さ付けを行い，同時に高さから勾配ベクトルを計算することで法線マップに変換する．これにより，ユーザは法線マップを意識することなく，削る・盛るなどの簡単な操作で，法線マップを生成することが可能となる．また，本システムでは，削る・盛る強さを表現するため，入力デバイスにタブレットを用い，筆圧機能により力の加減を行う．

3. システム構築

3.1 システム構成

本システムの構成を**図 1** に示す．本システムは，3D アプリケーション，ディスプレイ，およびタブレットで構成され，3D アプリケーションは主に，

- GUI 部
- オブジェクト管理部
- カメラ管理部
- ライト管理部
- マテリアル管理部
- ファイル入出力部

で構成される．GUI 部では，ウィンドウ，ボタンなどの各種 GUI コンポーネントの管理を行う．オブジェクト部では，面，エッジ，頂点，および UV 座標の管理を行う．マテリアル部では，ライトを当てたときの各種反射係数やテクスチャなどの管理を行う．ファイル入出力部では，Cg シェーダプログラムの入力，および MQO ファイル，各種テクスチャの入出力を行う．本システムでは，ポインティングデバイスにワコムのタブレット intuos i-900 を用い，筆圧により，削る・盛るサイズまたは強さの調整を行う．タブレットの筆圧情報を取得するため，LCS/Telegraphics が開発したタブレット用 API である Wintab を用いた．法線マッピングを行うには，ジオメトリの各頂点の法線ベクトルおよび UV マップされたジオメトリの各面

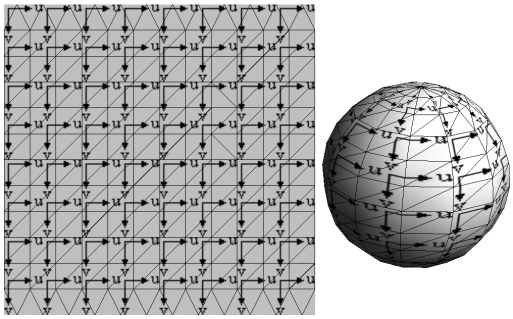


図 2 UV マッピング
Fig. 2 UV mapping.

上の点に対応する法線マップの RGB 値をもとにライティングを行う必要がある．そのため，本システムでは，シェーダ言語に C for graphics(Cg)[2),3),21)] を用い，法線マッピング用のシェーダプログラムを Cg コンパイラで OpenGL 形式に変換し，OpenGL を介して GPU に転送を行い，フラグメント単位でライティングを行う．

4. 仮想彫刻に基づく動的法線マップ生成

法線マップは 2 次元のテクスチャであり，3D オブジェクトの形状および通常貼り付けるテクスチャも考慮して作成する必要がある．そのため，3D モデリングとテクスチャマッピング（UV 展開およびテクスチャの用意）が済んだ状態で法線マップを作成しないと，モデル修正またはテクスチャ修正のたびに再度作り直しとなり，手間がかかってしまう．本章では，あらかじめユーザが 3D オブジェクトおよびテクスチャを用意し，**図 2** のように，UV 展開された状態として話を進める．また，前処理として，3D オブジェクトおよびテクスチャを読み込む際，プログラム側でテクスチャと同じサイズの RGB(127, 127, 127) で初期化された高さマップおよび RGB(127, 127, 255) で初期化された法線マップを自動作成しておく．

4.1 処 理 手 順

3D オブジェクトに対して直接彫刻を行うには，タブレットのドラッグ操作時に，高さマップおよび法線マップを書き換え，書き換えた法線マップをもとに，フラグメント単位でレンダリングを行う処理を随時行っていく（**図 3** 参照）．まず，3D オブジェクトの表面に対してピッキング処理を行い，ヒットしたときのウィンドウ座標およびヒットした面に属する頂点のウィンドウ座標から内分比を求める．次に，内分比およびピックされた面に属する頂点の UV 座標から，ピックされた面上の点の UV 座標の計算を行う．次に，UV 座標

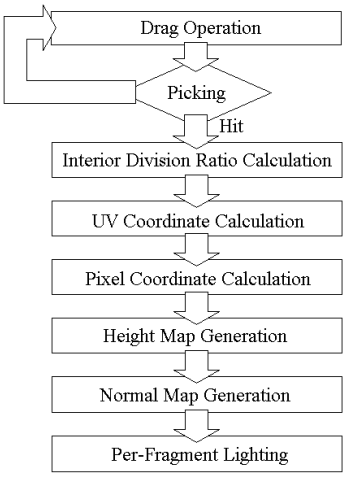


図 3 処理手順
Fig. 3 Procedure.

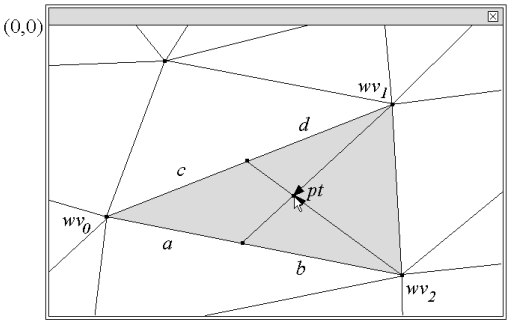


図 4 ウィンドウ座標
Fig. 4 Window coordinate.

からピクセル座標の計算を行い，ピクセル座標，筆圧，ブラシ形状などから，高さマップの書き換えを行う．最後に，書き換えたピクセル周りの高さから勾配ベクトルを計算し，法線マップを書き換えていく．書き換わった法線マップはフラグメント単位でライティングされ，擬似的な凹凸としてリアルタイムにレンダリングされる．

4.2 ピクセル座標計算

ピッキング処理で面がヒットした場合，ピックされたときのウィンドウ座標 pt およびヒットした面の位置ベクトル $\mathbf{v_0}$，$\mathbf{v_1}$，$\mathbf{v_2}$ を取得する．次に，位置ベクトル，ビューポート，モデルビュー行列，およびプロジェクション行列から gluProject[22)] で，$\mathbf{v_0}$，$\mathbf{v_1}$，$\mathbf{v_2}$ のウィンドウ座標 wv_0，wv_1，wv_2 をそれぞれ取得する．次に，$(p_t - wv_1)$，$(wv_2 - wv_0)$ から，エッジ wv_0wv_2 の内分比 $a : b$ を求め，同様に $c : d$ を求める（**図 4** 参照）．次に，ピックされた面に属する頂点の UV 座標 uv_0，uv_1，uv_2 を取得し，求めた内分

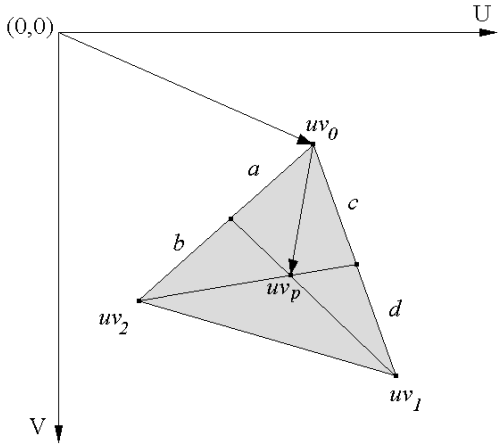


図 5 UV 座標計算
Fig. 5 UV coordinate calculation.

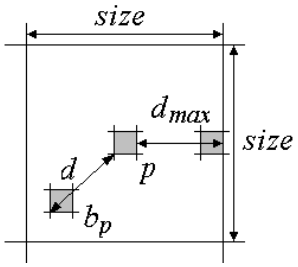
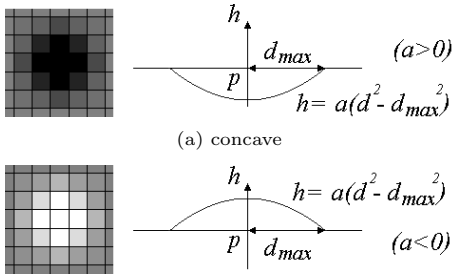


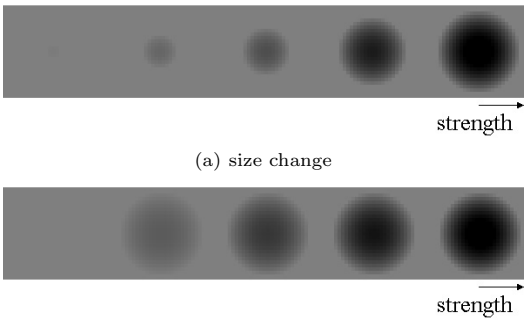
図 6 ブラシサイズ
Fig. 6 Brush size.



(a) concave

(b) convex

図 7 円形ブラシ
Fig. 7 Circular brush.



(a) size change

(b) depth change

図 8 筆圧を用いた円形ブラシ
Fig. 8 Circular brush using pen pressure.

比から，$uv_p - uv_0$ を求める．

$$uv_p - uv_0 = \frac{ad(uv_2 - uv_0)}{ad + bc + bd} + \frac{bc(uv_1 - uv_0)}{ad + bc + bd} \quad (1)$$

ピックされた面上の UV 座標 $uv_p(x_{uv_p}, y_{uv_p})$ は，

$$uv_p = uv_0 + (uv_p - uv_0) \quad (2)$$

で求まる（**図 5** 参照）．テクスチャの幅を w，高さを h としたとき，ピクセル座標 $p(x_p, y_p)$ は以下の式で求まる．

$$x_p = w * x_{uv_p}$$
$$y_p = h * y_{uv_p} \quad (3)$$

ヒットした面が 4 角ポリゴン以上の場合，3 角ポリゴンに分割して同様の処理を行う．

4.3 高さマップの書き換え

前節で求めたピクセル座標 p 周りのピクセルに対して高さマップの書き換えを行う．p 周りの範囲はブラシサイズによって変化する（**図 6** 参照）．ここで，ブラシサイズはユーザが任意に設定可能とする．

4.3.1 ブラシ生成

たとえば**図 7** のような円形ブラシを作成する場合，ブラシの範囲内の点 b_p と p との距離 d に応じて高さを変化させる．p から水平（垂直）方向への最大距離を d_{max} としたとき，ブラシの範囲内の点 b_p における高さ h を以下の式で求める．

$$h = a(d^2 - {d_{max}}^2) \quad (4)$$

ここで，a は係数でユーザが任意に設定可能とする．また，削る場合は正の値をとり，盛る場合は負の値をとる．

b_p における書き換え前の高さを H_{old}，色の濃さを t としたとき，H_{new} を以下の式で求め，高さマップを書き換える．

$$H_{new} = H_{old} + h * t \quad (5)$$

ここで，t はユーザが任意に設定可能とする．

タブレットの筆圧を用いる場合，筆圧の強さに応じてブラシサイズまたは色の濃さ t を変化させる（**図 8** 参照）．

4.4 法線マップの書き換え

書き換わった高さマップをもとに，ブラシサイズより 1 ピクセル周り広い範囲（**図 9** 参照）で法線マップの書き換えを行う．この範囲内における点を a_p としたとき，a_p の高さ H_g，a_p の U 方向に 1 ピクセ

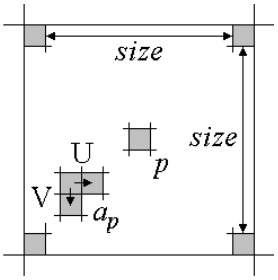
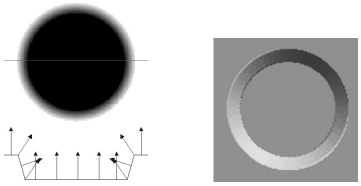


図 9 有効範囲
Fig. 9 Effective area.



(a) normal calculation (b) normal map

図 10 法線マップ生成
Fig. 10 Normal map creation.

ルの高さ H_r，および a_p の V 方向に 1 ピクセルの高さ H_a から法線ベクトル $\mathbf{v_{a_p}}(x_{v_{a_p}}, y_{v_{a_p}}, z_{v_{a_p}})$ を求める．

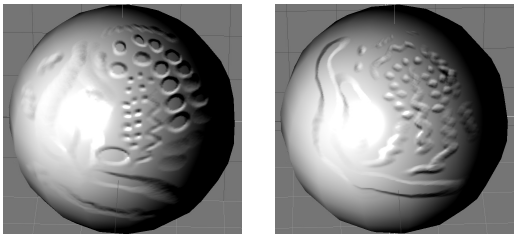
$$\mathbf{v_{a_p}} = \frac{(H_g - H_r, H_g - H_a, g)}{|\,(H_g - H_r, H_g - H_a, g)\,|} \tag{6}$$

ここで，g は勾配の強さで，この値が大きいほどなだらかな法線マップが得られる．最後に，この $\mathbf{v_{a_p}}$ を以下の式で RGB 値 $n(r_n, g_n, b_n)$ に変換し，法線マップを書き換える（**図 10** 参照）．

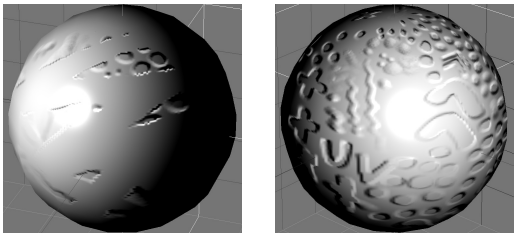
$$\begin{aligned} r_n &= 255 * (0.5 * x_{v_{a_p}} + 0.5) \\ g_n &= 255 * (0.5 * y_{v_{a_p}} + 0.5) \\ b_n &= 255 \end{aligned} \tag{7}$$

4.5 レンダリングおよび外部出力結果

書き換えた法線マップ，テクスチャ，マテリアルの各種パラメータ，3D オブジェクトの幾何データなどをもとに，フラグメント単位で法線マッピングおよびライティングを行う．これにより，タブレットでドラッグした部分が擬似的な凹凸としてレンダリングされる．**図 11** に，削る，盛る，平坦化ブラシを用いて彫刻を施した各種レンダリング結果，**図 12** に，法線マップの各種出力画像を示す．法線マッピングは，ディスプレイスメントマッピングと違い，実際に幾何形状は変化しない．そのため，輪郭の凹凸表現はできず，また，極端に大きな凹凸を表現することはできない．図 11 (a), (b), (d) では，局所的に最大筆圧で彫刻を行っており，法線マッピング単体で表現可能な最大高さで法線ベクトルを変化させている．このため，他のマッピング手法と組み合わせる等しない限り，図 11 の凹凸具合を超えた凹凸表現はできない．



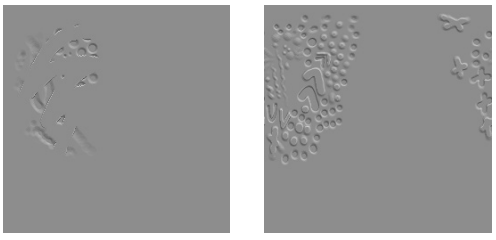
(a) sub (b) add



(c) flat (d) sub+add

図 11 各種レンダリング結果
Fig. 11 Rendering results.

(a) sub (b) add



(c) flat (d) sub+add

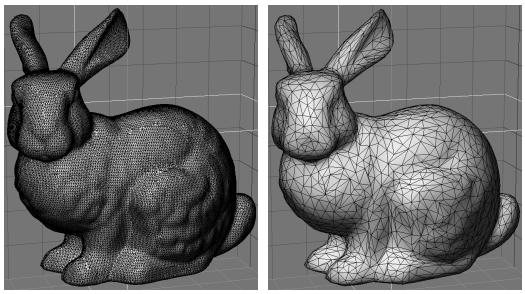
図 12 各種出力結果
Fig. 12 Output results.

5. 品 質 比 較

提案手法の有効性を示すため，**表 1** に示す 4 つのモデルの最終的なレンダリング結果において品質比較を行った．**図 15** に彫刻を行っていく過程を示す．各テクスチャのサイズは 256×256，ブラシのサイズ *size* は可変，円形ブラシ（$a = 1.0$）を用い，筆圧の強さに

表 1 比較条件
Table 1 Comparison condition.

	model	vertex number	polygon number	normal mapping
model 1	hi-polygon	34834	69451	off
model 2	low-polygon	2077	3999	off
model 3	low-polygon	2077	3999	on (baked normal)
model 4	low-polygon	2077	3999	on (proposal technique)



(a) hi-polygon model (b) low-polygon model

図 13 テスト用モデル
Fig. 13 Test models.

応じて色の濃さ $t[0, 1.0]$ を変化させている．法線マップ生成の際の勾配の強さ g は 100 としてある．マテリアルは，光の反射具合を分かりやすくするため，鏡面光に対する色成分と鏡面に対する指数を最大に設定してある．ライティングは，フォンシェーディング用シェーダプログラムを法線マッピング用に拡張したものを用い，フラグメント単位でシェーディングを行っている．また，使用した CPU は Pentium4 2.8 GHz，ビデオカードは FireGL T2-128 である．タブレットによるドラッグ操作の際，特に動作が重いといったことはなく，リアルタイムに彫刻を施すことができた．図 15 を見ると，徐々に彫刻が施され，凹凸がついていく過程が見てとれる．

図 16 は，提案手法とのレンダリング比較結果である．図 16 (a) は，**図 13** (a) と同じハイポリゴンモデル，図 16 (b) は，図 13 (b) と同じローポリゴンモデル，図 16 (c) は，図 13 (a) のジオメトリの法線ベクトルを図 13 (b) の UV マップ（**図 14** 参照）に焼き込み，法線マッピングを行ったローポリゴンモデル，図 16 (d) は，提案手法を用いて図 13 (b) に対して彫刻を行ったローポリゴンモデルである．図 16 (b) と比べ，図 16 (c)，図 16 (d) ともに，より精細なレンダリング結果が得られていることが分かる．提案手法では，図 16 (c) に比べ，よりシャープな結果を得ることができた．また，提案手法ではオブジェクトに対して直接彫刻を行っていくため，図 16 (a) に似せた結果を得るのはデザイナの腕次第である．

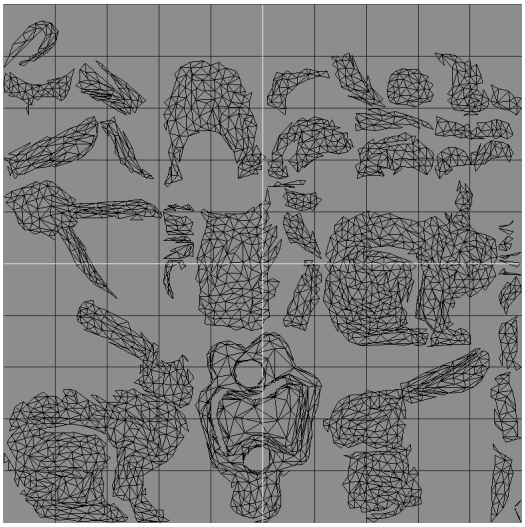


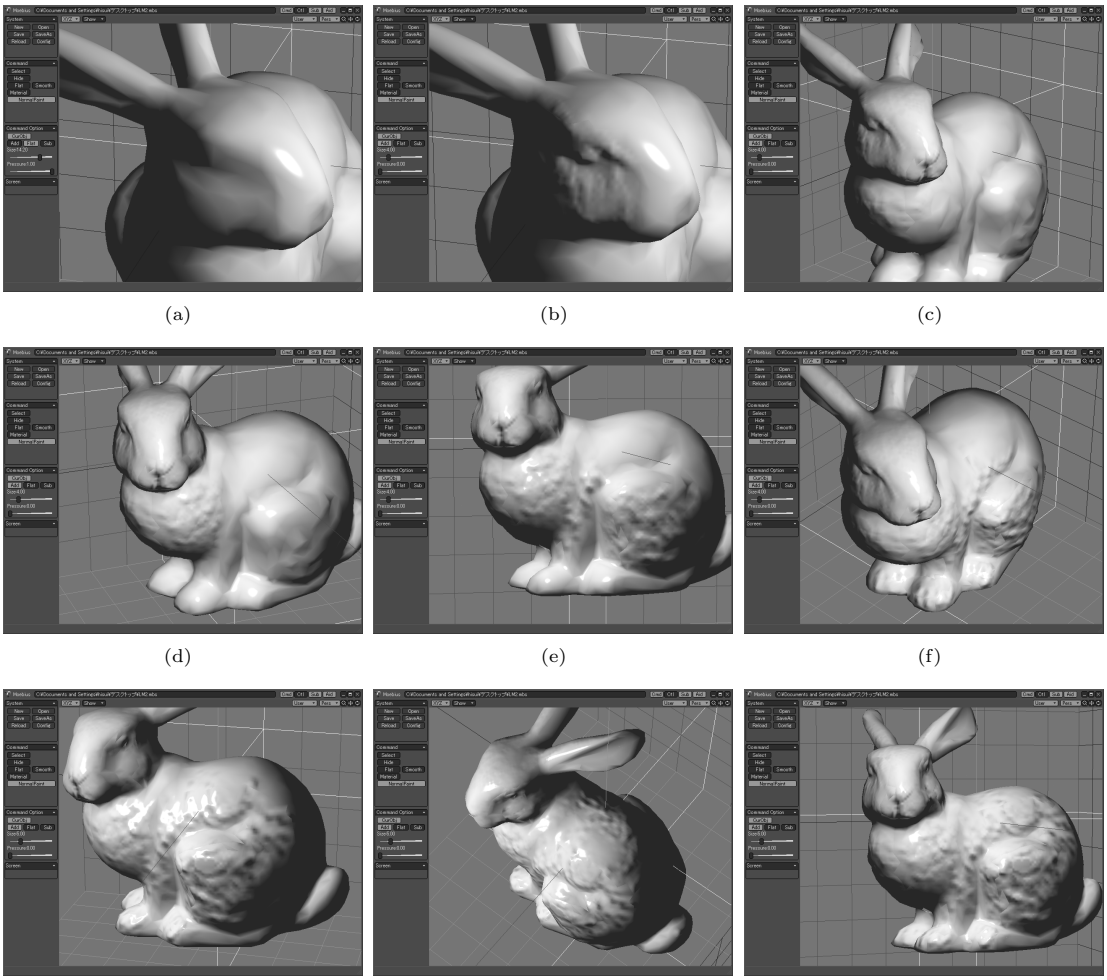
図 14 UV 展開図
Fig. 14 UV development.

凹凸具合を修正する場合，平坦ブラシで消して，彫刻し直せばよいため，ハイポリゴンモデルを修正してローポリゴンモデルの法線マップを焼き込み直す方法に比べ，低コストで柔軟かつ即座に対応可能である．また，提案手法では，テクスチャマッピングされた状態で彫刻が行えるため，テクスチャに合わせて凹凸具合を確認しながら法線マップを生成することも可能である．

図 17 に，PNG 形式で外部出力した高さマップおよび法線マップを示す．高さマップにおいて，色が濃いところは，強い筆圧で彫刻を行った部分，薄いところは，弱い筆圧で彫刻を行った部分である．法線マップにおいて，筆圧の強い部分は色が大きく変化し，弱い部分は小さく変化していることが分かる．

6. おわりに

本研究では，仮想彫刻に基づき，リアルタイムに法線マップを生成可能なシステムの開発を行った．具体的には，法線マッピングが最終的に擬似的な凹凸としてレンダリングされることに注目し，仮想彫刻に基づき，3D オブジェクトを直接削る・盛ることにより，インタラクティブな操作で直感的に法線マップを生成可



(a) (b) (c)

(d) (e) (f)

(g) (h) (i)

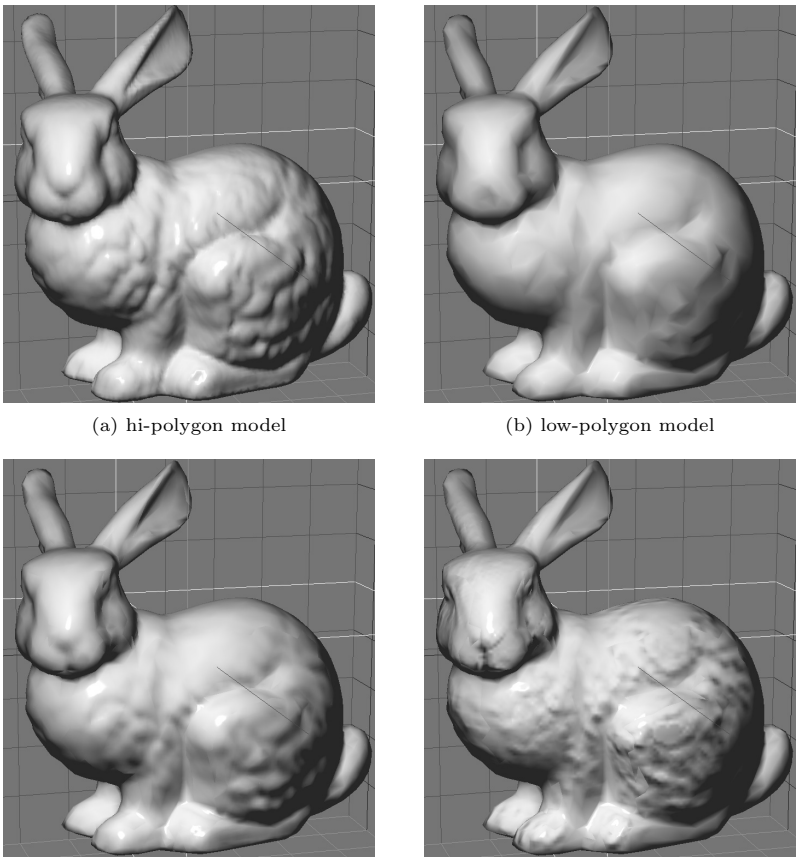
図 15 彫刻の過程
Fig. 15 Sculpting process.

能なシステムの開発を行った．また，本システムを用いることにより，ユーザは法線マップを意識することなく，リアルタイムかつインタラクティブに法線マップを生成可能なことを確認した．

今後の課題としては，システムをさらに使いやすいものとするために，GUI の強化，UV 編集機能や自動 UV 展開機能の実装，様々なブラシの作成などがあげられる．また，さらなる発展として，他のマッピング手法との組合せによる，より精細な擬似凹凸表現，3D ポインタによる仮想ブラシおよび力覚提示装置によるスカルプティング，擬似的な凹凸へのインタラクション，ハプティックテクスチャマッピングによるオブジェクト表面の質感表現などがあげられる．

参 考 文 献

1) 宮田一乗，高橋誠史，黒田 篤：GPU コンピューティングの動向と将来像，芸術科学会論文誌，Vol.4, No.1, pp.13–19 (2005).
2) Fernando, R. and Kilgard, M.J.: *The Cg Tutorial: The Definitive Guide to Programmable Real-Time Graphics*, p.384, Addison-Wesley (2003).
3) Watt, A.H. and Policarpo, F.: *Advanced Game Development With Programmable Graphics Hardware*, p.384, A.K. Peters Ltd. (2005).
4) Blinn, J.F.: Simulation of wrinkled surfaces, *Computer Graphics* (*Proc. SIGGRAPH'78*), Vol.12, No.3, pp.286–292 (1978).
5) 尾上耕一，西田友是：風紋・砂丘を含む砂漠景観の表示法，電子情報通信学会論文誌 D，Vol.J86-

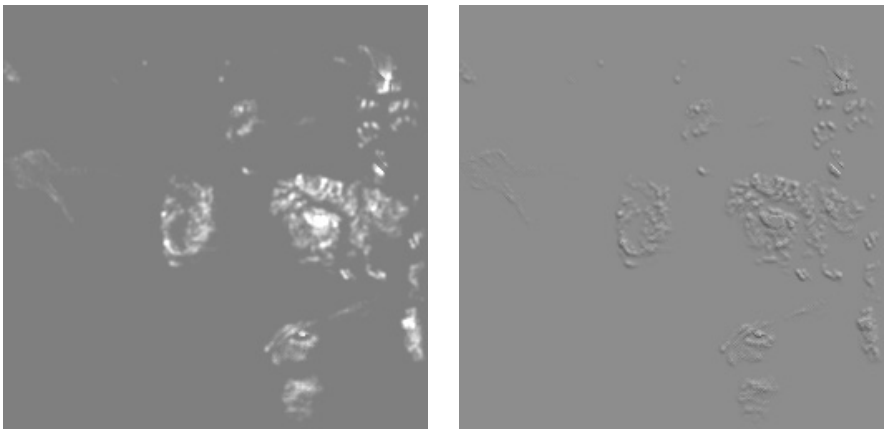


(a) hi-polygon model (b) low-polygon model

(c) low-polygon model (baked normal) (d) low-polygon model (proposal technique)

図 16 レンダリング比較
Fig. 16 Rendering comparison.

D2, No.2, pp.282–289 (2003).
6) Cock, R.L.: Shade trees, *Computer Graphics* (*Proc. SIGGRAPH'84*), Vol.18, No.3, pp.223–231 (1984).
7) Kaneko, T., Takahei, T., Inami, M., Kawakami, N., Yanagida, Y., Maeda, T. and Tachi, S.: Detailed Shape Representation with Parallax Mapping, *Proc. ICAT2001* (*11th International Conference on Artificial Reality and Tele-Existence*), Japan, pp.205–208 (2001).
8) Oliveira, M.M., Bishop, G. and McAllister, D.: Relief Texture Mapping, *Computer Graphics* (*Proc. SIGGRAPH'00*), pp.359–368 (2000).
9) Wrotek, R., Rice, A. and McGuire, M.: Real-Time Bump Map Deformations, *Poster and ACM SRC Talk at SIGGRAPH 2004*, Los Angeles (2004).
10) Wrotek, R., Rice, A. and McGuire, M.: Real-Time Collision Deformations Using Graphics Hardware, *Journal of Graphics Tools*, Vol.10, No.4, pp.1–22 (2005).
11) Yamasaki, T., Hayase, T. and Aizawa, K.: Mathematical Error Analysis of Normal Map Compression Based on Unity Condition, *IEEE Int. Conf. on Image Processing* (*ICIP2005*), Genova (2005).
12) Wang, Y., Fröhlichi, B. and Göbel, M.: Fast Normal Map Generation for Simplified Meshes, *Journal of Graphics Tools*, Vol.7, No.4, pp.69–82 (2002).
13) Gumbau, J., González, C. and Chover, M.: Fast GPU-based normal map generation for simplified models, *Proc. WSCG2006 Posters*, Vol.80-86943-04-6, Plzen, Czech Republic (2006).
14) Mizuno, S., Kobayashi, D., Okada, M., Toriwaki, J. and Yamamoto, S.: Creating a Virtual Wooden Sculpture and a Woodblock Print with a Pressure Sensitive Pen and a Tablet, *FORMA – J. of Society for Science on Form*, Vol.21, No.1, pp.49–65 (2006).
15) Cristiano, S., Fiorentino, M., Monno, G. and



(a) height map (b) normal map

図 17 出力結果
Fig. 17 Output results.

Uva, A.E.: Real-Time Particle Based Virtual Sculpturing, *Proc. ADMAIAS'04 Conference*, Vol.88-900637-2-6, Italy (2004).
16) Jagnow, R. and Dorsey, J.: Virtual Sculpting with Haptic Displacement Maps, *Proc. Graphics Interface*, pp.125–132, Canada (2002).
17) Kang, S.B.: Depth Painting for Image-based Rendering Applications, Tech. Rep. CRL, Compaq Cambridge Research Lab. (1998).
18) Oh, B.M., Chen, M., Dorsey, J. and Durand, F.: Image-Based Modeling and Photo-Editing, *Computer Graphics* (*Proc. SIGGRAPH'01*), pp.433–442 (2001).
19) Okabe, M., Zeng, G., Matsushita, Y., Igarashi, T., Quan, L. and Shum, H-Y.: Single-View Relighting with Normal Map Painting, *Proc. Pacific Graphics*, pp.27–34 (2006).
20) Wu, T-P., Tang, C-K., Brown, M. S. and Shum, H-Y.: ShapePalettes: Interactive Normal Transfer via Sketching, *Computer Graphics* (*Proc. SIGGRAPH'07*), In Press (2007).
21) 金谷一郎：3D-CG プログラマーのためのリアルタイムシェーダー理論と実践—「古典的ライティング・モデル」から「グローバル・イルミネーション」まで，p.180, 工学社 (2004).
22) Woo, M., Neider, J., Davis, T. and Shreiner, D.: *OpenGL Programming Guide: The Official Guide To Learning OpenGL, Version 2*, p.896, Addison-Wesley (2005).

(平成 19 年 4 月 2 日受付)
(平成 19 年 9 月 3 日採録)

脇田 航

平成 16 年愛媛大学工学部情報工学科卒業．平成 18 年同大学大学院理工学研究科情報工学専攻博士前期課程修了．現在，同大学院理工学研究科電子情報工学専攻博士後期課程在学中．3DCG，バーチャルリアリティ等の研究に従事．

井門 俊（正会員）

平成 8 年東京工業大学大学院総合理工学研究科博士後期課程単位取得退学．現在，愛媛大学大学院理工学研究科講師．画像符号化，バーチャルリアリティ等の研究に従事．博士（工学）．