

# 点で表現された曲面の穴埋めのための 効率的なアップサンプリング法

金森 由博<sup>†</sup> 高橋 成雄<sup>††</sup> 西田 友是<sup>††</sup>(正会員)

<sup>†</sup> 東京大学大学院情報理工学系研究科

<sup>††</sup> 東京大学大学院新領域創成科学研究科

あらまし 点集合による形状表現は、従来の多角形メッシュと異なり、点同士の接続情報を用いずに大規模な 3 次元形状データを扱うことができるため、CG の分野において非常に有望とされている表現方法のひとつである。点集合で表現された形状を描画するためには、通常サーフェルと呼ばれる円盤が広く用いられているが、サンプリング密度が不十分な領域では穴が生じてしまう。

これを解決するため、本論文ではサーフェルを用いて曲面を表現する際に穴が生じないように、アップサンプリングを行う手法を提案する。穴を埋めるために、提案法は各サーフェルの周囲の隙間を検出し、サーフェルを追加して隙間を埋める。提案法はサーフェルの重なりとユーザが指定したサーフェルの半径を考慮することによって、追加するサーフェルの数を少なく抑える。実験により、提案法が多重解像度 (level-of-detail) の制御や対話的なモデリングなどに広く応用できることを示す。

キーワード：点群処理, 穴埋め, アップサンプリング

**Summary** Point-sampled geometry has become one of the most promising representations in computer graphics because, unlike conventional polygonal meshes, it can manage large-scale 3D data without accounting for the point connectivity. The common rendering technique for point-sampled geometry is to introduce a set of oriented disks called *surfels*, however, it still suffers from unexpected holes on insufficiently sampled regions.

This paper presents a novel up-sampling method for filling such holes on the surfaces consisting of surfels. For filling the holes, the present method extracts gaps around each surfel and inserts new surfels to fill the gaps. This requires a small number of surfels in our framework by taking into account their overlaps together with the user-specified radii of surfels. Several experiments have been conducted to demonstrate that the present method has wide applications such as level-of-detail control and interactive editing of point-sampled geometry.

**Key words:** point-sampled geometry, hole-filling, up-sampling

## 1. はじめに

点集合による形状表現は、従来の多角形メッシュによる表現と違い、頂点間の接続情報を用いずに 3 次元形状を扱えるため、大規模なデータの効率的な表現方法とし

て有望視されている。点集合で表現された形状を描画するためには、通常「サーフェル」と呼ばれる円盤が広く用いられている<sup>6)</sup>。しかし、点のサンプリング密度が不十分な領域では穴が生じてしまう。このような穴を埋めるために点を増やす手法が多く提案されている。代表的なものとして Voronoi 図を用いた手法<sup>1)</sup>、メッシュの細分割を応用した手法<sup>3)</sup>があるが、穴を覆うために比較的多くのサーフェルを必要としてしまう。

本稿では、既存手法より少ないサーフェルで穴埋めを行うアルゴリズムを提案する。提案法は各サーフェルの

“An Efficient Up-Sampling Method for Filling Holes on Point-sampled Surfaces” by Yoshihiro KANAMORI, (Graduate School of Information Science and Technology, the University of Tokyo), Shigeo TAKAHASHI, and Tomoyuki NISHITA, (Member), (Graduate School of Frontier Sciences, the University of Tokyo).

周囲を検査することで自動的に穴の検出・サーフェルの追加を行う。サーフェルの半径は一定あるいは曲率に応じて可変にする（曲がった箇所は比較的密に、平らな箇所は比較的疎にする）ことができ、サーフェルの分布を局所的に均一に保つことができる。

## 2. 関連手法

本節では、点集合によって表現された形状のためのアップサンプリング法について、関連手法を述べる。

Stamminger ら<sup>9)</sup>はパラメータ空間でアダプティブに曲面をサンプリングする手法を提案した。彼らの手法は描画時に十分なだけのサンプルが得られるまで単純な規則で各点の周りに点を追加するというもので、穴の検出はできない。Alexa ら<sup>1)</sup>は、局所的に最小二乗法で当てはめた平面へ点を投影し、最大空円問題<sup>5)</sup>を反復的に解くことでアップサンプリングを行った。しかし、彼らはサーフェルの重なり具合を考慮しておらず、曲面を覆うために必要以上にサーフェルを追加してしまう。

Guennebaud ら<sup>2)</sup>は細分割曲面の考えを応用し、局所的に点の隣接関係を構築して点を増やす手法を提案した。これは点の分布が均一な場合にしか適用できなかったが、不均一な場合や大きな穴に対応するため、隣接関係の構築や点の追加の方法に改良がなされた<sup>3)</sup>。しかし、改良された手法ではサーフェルの重なりが非常に大きく、大きな穴を自動的に検出することができない。さらに、いずれの手法でも細分割曲面同様、点数は整数倍ずつしか増やすことができないため、サンプリング密度の調整は柔軟でない。

多くのアップサンプリング手法では、サンプリング密度を制御するために大域的最適化が用いられる。代表的なものとしてパーティクルシミュレーション<sup>8), 10)</sup>が挙げられる。これは各点に対して粒子間力を定義し、平衡状態になるまで反復計算して点の分布を均一にするというものである。しかしパーティクルシミュレーションは計算的に安定でなく、計算コストが非常に高いため大規模なデータには向かない。

提案法はサーフェルの半径を考慮し、サーフェルの重なりをできるだけ小さく抑え、点の局所的な分布を均一に保つ。サンプリング密度はサーフェルの半径を指定することで柔軟に制御することができる。

## 3. 穴の検出と穴埋め

### 3.1 提案法の概要

本稿において「穴」とは表面形状でサーフェルに覆われていない領域のことと定義する。入力となる表面形状は閉じており、元々穴がないと仮定する。提案法は、サー

フェルで表現された形状データを入力とし、穴埋めを行う。出力されるデータは、ユーザが予め指定したサーフェルの半径で穴が生じないことが保証されている、アップサンプリングされた点群である。以下、入力データおよび出力データのことを、それぞれ入力点集合および出力点集合と呼ぶことにする。各サーフェルは真円の（短軸と長軸の長さが等しい）円盤であり、面の向きを表す単位法線ベクトルと表面の色をもつものとする。

提案法は次の3ステップからなる（図1参照）：各サーフェル  $i$  について、

1. サーフェル  $i$  の近傍のサーフェルを、サーフェル  $i$  の中心を通る接平面に投影
2. サーフェル  $i$  の周囲の穴を検出し、新しいサーフェルを追加して穴を埋める
3. 接平面で新しく追加したサーフェルを、近傍の点群が表現する曲面に逆投影

この穴埋め手続きは、各サーフェルの周囲の穴が近傍のサーフェルによって完全に覆われたとき終了する。

### 3.2 穴の検出と穴埋め

提案法は、穴の検出を簡便にするため、注目するサーフェルによって定義される、近傍の点群が表現する曲面の接平面を用いる。以下では、まず2次元の場合におけるアップサンプリングのアルゴリズムを述べ、続いてそれを3次元の場合に拡張する。

#### 3.2.1 基本的な考え（2次元の場合）

提案するアップサンプリングアルゴリズムは、「穴の検出」および「穴埋め」の2つの手続きからなる。穴の検出手続きは、各サーフェルの周囲の穴を検出し、新しいサーフェルが必要かどうかを決定する。そして穴埋め手続きは、新しいサーフェルを配置して穴を埋める。過剰にサーフェルを追加せず、サーフェルの分布を均一にするため、提案手法では穴の検出時には穴を正確に検出し、穴埋め時にはサーフェルの重なりを減らし、近傍のサーフェルの分布を均一に保つようになっている。

仮定：2次元の平面上に  $N$  個のサーフェルが不均一に分布しているとする。各サーフェル  $i$  ( $0 \leq i < N$ ) の中心は  $\mathbf{p}_i$ 、半径は  $r_i$  ( $r_{min} \leq r_i \leq r_{max}$ ) とする。もし2つのサーフェルが完全に一致する場合は、冗長なので一方を除去する。サーフェルの半径は急激には変化しないものとする。

穴の検出：各サーフェル  $i$  の周囲の穴を検出するため、サーフェル  $i$  の境界が近傍のサーフェルによって覆われているかを検査する。サーフェル  $i$  とサーフェル  $i'$  とが交差するとき、すなわち  $|r_i - r_{i'}| < \|\mathbf{p}_i - \mathbf{p}_{i'}\| < r_i + r_{i'}$

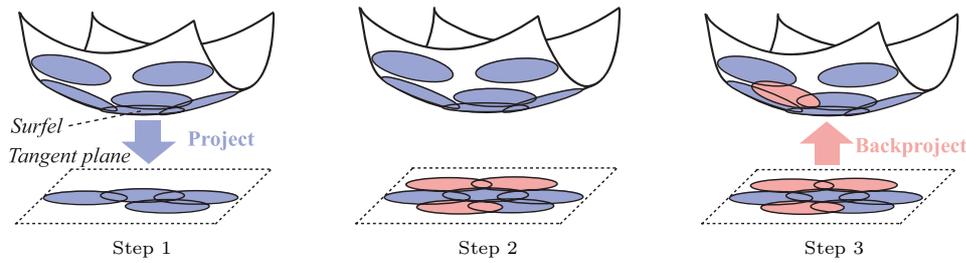
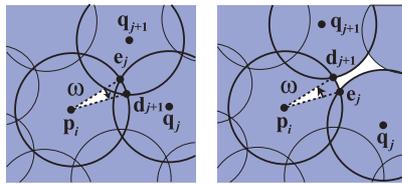


図 1 提案法の概要. Step 1: 各サーフェル (青) を接平面に投影, Step 2: 接平面上で穴を検出し, 埋める. Step 3: 新しく追加したサーフェル (ピンク) を点群が表現する曲面に逆投影する. 説明のため, 図では曲面と接平面は離して描いている.

Fig. 1 Overview of the present method. Step 1: project existing surfels (in blue) onto a tangent plane, Step 2: detect and fill gaps on a tangent plane, Step 3: backproject new surfels (in pink) onto the point-sampled surface. The tangent plane is separated from the point-sampled surface for an illustrative purpose.



(a) 穴なし ( $\omega \leq 0$ ) (b) 穴あり ( $\omega > 0$ )

図 2 穴の検出.

Fig. 2 Hole-detection.

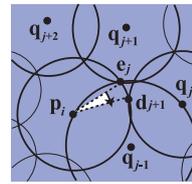


図 3 穴の検出手続きの例外.

Fig. 3 An exception of hole-detection.

て, 穴を覆うかどうかを検査する.

となるとき, サーフェル  $i$  の境界はサーフェル  $i'$  によって部分的に覆われる. それゆえ, 穴の検出手続きではサーフェル  $i$  と交差する近傍サーフェル ( $K$  個とする) を探索する. サーフェル  $j$  ( $0 \leq j \leq K$ , 中心は  $q_j$ , ただし  $q_0 = q_K$  とする) を, 交差するサーフェルのひとつとする. 各サーフェル  $j$  は反時計回りにソートしておく.

サーフェル  $i$  の周囲の穴は次のようにして検出する. サーフェル  $i$  と  $j$  の境界の交点を,  $p_i$  を中心として反時計回りに  $d_j$  および  $e_j$  とする. サーフェル  $i, j, j+1$  の間の穴を検出するため, 角度  $\omega = \angle e_j p_i d_{j+1}$  ( $-2\pi < \omega \leq 2\pi$ ) を計算する (図 2 参照). もし  $\omega \leq 0$  であれば, サーフェル  $i, j, j+1$  は互いに重なるので, それらの間に穴はない (図 2(a)). もし  $\omega > 0$  であれば穴がある (図 2(b)). サーフェル  $i$  の周囲に交差するサーフェルがない場合は  $\omega = 2\pi$  とする.

しかしながら, 上記の手続きだけでは実際には埋まっている穴を誤って検出する場合がある. 図 3 の例では, サーフェル  $j-1$  が, サーフェル  $i, j, j+1$  の間の穴を埋めている. このため, サーフェル  $i, j, j+1$  だけでなく近傍他のサーフェルが穴を覆うかどうかを調べる必要がある. 提案法ではサーフェルの半径が急激には変化しないと仮定して, サーフェル  $j-1$  および  $j+2$  のみについ

穴埋め: サーフェル  $i, j, j+1$  の間に穴がある場合 (すなわち  $\omega > 0$ ), 穴埋め手続きによって新しいサーフェルを追加し穴を埋める. 追加されるサーフェルの半径はすべて等しいものとし, それを  $r_{new}$  とする. サーフェルの重なりを減らすため, 次の式で計算される角度  $\Delta\omega$  を用いて, 等間隔に新しいサーフェルを追加することを考える:

$$\Delta\omega = 2 \arcsin \frac{-r_{new}^2 + r_{new} \sqrt{r_{new}^2 + 8r_i^2}}{4r_i^2} \quad (1)$$

ここで  $r_i$  はサーフェル  $i$  の半径である. この値は図 4 に示すような条件を満たすよう選んだ: 仮に, サーフェル  $i$  の近傍に他のサーフェルがなくサーフェルの半径が一定, すなわち,  $\omega = 2\pi$  かつ  $r_{new} = r_i$  とすると,  $\Delta\omega = \pi/3$  となり, 新しいサーフェルは中心点が六角形格子を成すように追加され, この場合の最適配置となる. 実験では  $r_{new} \neq r_i$  である場合にも良好な結果が得られた.

穴の広がり角度  $\omega$  および  $\Delta\omega$  とを考慮して, 新しいサーフェル  $k$  を追加する. このとき, すでに存在するサーフェルを動かしてしまうと, そのサーフェルの周りに穴が開く恐れがあり, 穴の検出処理が必要となる. これを避けるため, すでに存在するサーフェルは動かさないものとし, 必要に応じて新しいサーフェルをずらす. この手続

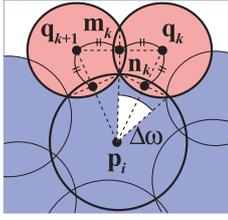


図 4  $\Delta\omega$  の導出.  $\Delta\omega$  はサーフェル  $k$  および  $k+1$  の中心点が次の条件を満たすように決定される:  $\|q_{k+1} - m_k\| = \|q_k - m_k\| = \|q_k - n_k\|$ .

Fig. 4 An illustration for  $\Delta\omega$ .  $\Delta\omega$  is determined so that the centers of new surfels  $k$  and  $k+1$  meet the condition:  $\|q_{k+1} - m_k\| = \|q_k - m_k\| = \|q_k - n_k\|$ .

きは次のように行う:

$\omega > 0$  の間, 以下を繰り返す:

$\omega > \Delta\omega$  の場合 (図 5a):

新しいサーフェルを 1 つ仮に配置;

$\omega \leftarrow \omega - \Delta\omega$  として処理を繰り返す;

$\omega \leq \Delta\omega$  の場合:

穴の両側のサーフェルが交差する (穴が小さい) 場合

新しいサーフェルがある場合 (図 5b):

新しいサーフェルをずらして穴埋め終了;

新しいサーフェルがない場合 (図 5c):

3 つのサーフェルの中心点からなる

三角形の垂心にサーフェルを追加して終了;

それ以外の場合 (図 5d):

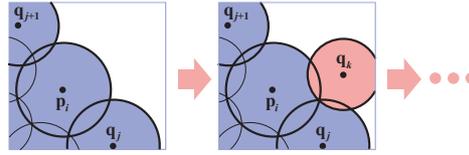
ちょうど穴が埋まるように

新しいサーフェルを 1 つ追加して終了;

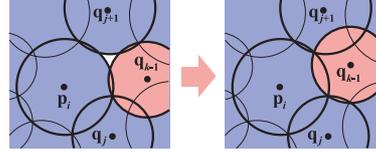
上記の手続きでは, 近傍の他のサーフェルを考慮していない. サーフェル  $k$  の中心点が, すでに存在するサーフェル  $l$  の半径内に入ってしまう, すなわち  $\|q_k - p_l\| < r_l$  となる場合がある (図 6). ここで  $p_l$  および  $r_l$  はそれぞれ, サーフェル  $l$  の中心点および半径である. このように 2 つのサーフェルが近づきすぎることを許すと, 全体として穴埋めに必要なサーフェルの個数が増えてしまう傾向がある. そこで, 新しいサーフェル  $k$  を線分  $q_k p_i$  に沿って, 次の条件を満たす位置までずらす:

$$\frac{\|q_k - p_i\|}{\|q_k - p_l\|} = \frac{r_i}{r_l}. \quad (2)$$

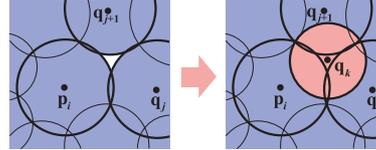
このようにしてずらせば, ずらすことによって  $p_i$  の周りに新たな穴ができることはない. この処理で中心点  $q_k$  がサーフェル  $i$  の内側に入ってしまった場合, サーフェル  $i$  の境界まで  $q_k$  を再度ずらす. なお図 6 の場合では, サーフェル  $k$  を点  $p_i$  に関してさらに半時計周りに移動させた方がサーフェル  $i$  の境界をより広く覆うことができる.



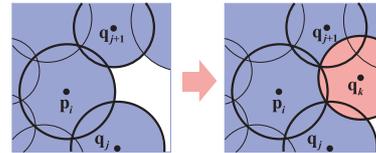
(a)  $\Delta\omega < \omega$  の場合.



(b) 穴が小さく, 新しいサーフェルがある場合.



(c) 穴が小さいが, 新しいサーフェルがない場合.



(d) 上記以外の場合.

図 5 穴埋め規則.

Fig. 5 The hole-filling rules.

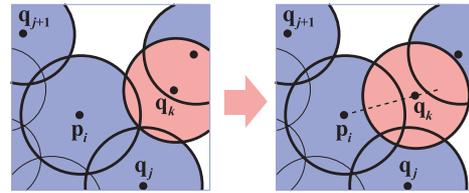


図 6 穴埋め規則の修正. サーフェル  $k$  の中心点  $q_k$  が近傍サーフェルの内側に入ってしまう場合,  $q_k$  を線分  $q_k p_i$  に沿って  $\|q_k - p_i\| / \|q_k - p_l\| = r_i / r_l$  となる位置までずらす.

Fig. 6 The modification rule for hole-filling. If  $q_k$ , the center of the new surfel  $k$ , is placed inside another existing neighbor,  $q_k$  is displaced along the segment  $q_k p_i$  so that  $\|q_k - p_i\| / \|q_k - p_l\| = r_i / r_l$  in order to keep the local sample distribution uniform.

しかし一般に, 半時計周りに移動させるとまた別のサーフェルと近づきすぎる場合があるため, 提案法ではこのような単純な処理に留めた.

この処理によって, 次に追加されるサーフェル  $k+1$  の位置もずれる. すなわち, サーフェル  $k+1$  はサーフェル  $i, k$  と 1 点で交わるように配置される.

既存手法との比較： 図 7 に、提案法と Alexa らの手法<sup>1)</sup>との比較を示す。彼らの手法では半径を制御する方法が提示されていないため、半径は一定として実験した。特に入力が疎な場合に、提案法は Alexa らの手法に比べ、より少ないサーフェルで領域を埋め尽くせることがわかる。

### 3.2.2 3次元への拡張

上記の手続きを 3次元の場合に拡張する。各サーフェルの半径が物体形状に比べ十分小さいと仮定すると、点でサンプリングされた曲面  $S$  は、局所的に点  $p_i$  を通る  $S$  の接平面  $T_i$  で近似できる。提案法では、サーフェル  $i$  の近傍のサーフェルを接平面  $T_i$  に投影することで、2次元の場合の手続きを用いて接平面  $T_i$  上で穴の検出および穴埋めを行う。投影されたサーフェル  $j$  の形は楕円となるが、計算の簡便のため、穴埋め手続きではこれを半径  $\max(0, n_j \cdot n_i) r_j$  の真円として扱う。ここで  $n_i$  と  $n_j$  はそれぞれ、サーフェル  $i, j$  の単位法線ベクトルである。

3次元の穴埋め処理は次のように行う：各サーフェル  $i$  について、その近傍サーフェルを  $p_i$  における接平面に投影する。そして接平面上で 2次元の穴の検出および穴埋め手続きを行う。最後に、新しく追加したサーフェルを曲面  $S$  に逆投影する。曲面  $S$  は滑らかで閉じていると仮定しているので、この手続きを繰り返せば穴埋めが完了する。

曲面への逆投影の計算： 接平面から曲面  $S$  への逆投影のために、提案法では移動最小二乗法 (*moving least squares*; 以下 MLS)<sup>4)</sup>を用いる。点でサンプリングされた曲面  $S$  を、Shen らの手法<sup>7)</sup>と同様に、2次元曲面で局所的に補間あるいは近似する。いずれの場合も、MLSによって計算される曲面は入力点集合によって定義される曲面であるため、入力点集合と出力点集合とは別に扱う。

サーフェルの半径の決定： 曲率に応じて半径を変える場合、提案法はサーフェル  $i$  の半径を曲率に基づいて決定する。具体的には、中心点  $p_i$  において計算した、近似 2次元曲面の主曲率の絶対値の最大値  $\kappa_i$ 、およびユーザ定義の値  $\epsilon_r$  を用いて次のように計算する： $r_i = \epsilon_r / \sqrt{\kappa_i}$ 。ここで、サーフェル  $i$  の周囲に追加されるサーフェルの半径は、各サーフェルの中心点での曲率から計算されることになるが、重なりを減らしつつ中心点の位置を決めるには、各サーフェルの半径が必要になる。そのため、提案法は新しいサーフェルの半径  $r_{new}$  を、各サーフェルの中心点においてではなく、サーフェル  $i$  の中心点  $p_i$  において計算する。これは曲率が滑らかに変化するという仮定に基づくものである。 $r'$  を新しいサーフェルの中心点において計算した半径のひとつとする。曲率の変化が激

しく  $r_{new}$  と  $r'$  との比が閾値を超える場合、提案法では  $r_{new}$  を  $r'$  で置き換え、穴埋めを再度行う。

## 4. 実験結果

C++および OpenGL を用いて実装を行った。以下の実験は、Pentium D 3.0GHz, 2GB RAM, nVIDIA GeForce 7800 GTX 搭載の PC 上で行った。また、MLS の計算のため入力点集合には *kd-tree*、穴の検出・穴埋めのため出力点集合には *grid* を用いて、それぞれ近傍探索を高速化した。

図 8 に、Guennebaud らの手法<sup>3)</sup>Alexa らの手法<sup>1)</sup>、提案法のそれぞれを用いて穴埋めを行った結果を示す。Guennebaud らの手法では、穴埋めを行う場合、サーフェルの半径を事前に穴を十分覆うように設定し、細分割を繰り返す必要がある。このため、われわれの実験では初期点集合は疎なものを用いた。Alexa らの手法および提案法について、サーフェルの半径は一定として比較した。半径の長さは、各形状のバウンディングボックスの対角線の長さを 1 とする。われわれの実験で用いた Guennebaud らおよび Alexa らの手法の実装はともに最適化されていないが、参考のために穴埋めにかかった計算時間を付記している。

比較的違いが見やすい sphere の例について、

- Phong シェーディングして描画した結果
- 環境マッピングによって直線を映した結果 (穴埋めされた曲面の品質を評価するため)
- サーフェルを一定倍率で縮小して描画した結果 (点の分布を示すため)

をそれぞれ示した。Guennebaud らの手法の結果 (図 8(a) および (d)) では環境マッピングされた線が歪んでおり、凹凸があることを示している。一方、Alexa らの手法 (図 8(b) および (e)) および提案法 (図 8(c) および (f)) では、サンプル数が増えるに従って滑らかな曲面が得られている。いずれの例においても、提案法によってアップサンプリングを行った場合に最も少ないサーフェル数に抑えられていることがわかる。

図 9 は、曲率を考慮してサーフェルの半径を変え、dragon モデルを再サンプリングした結果である。主曲率を考慮して半径を変えることにより、半径を一定にした場合に比べ、視覚的差異を生じることなくサーフェルの数を大幅に削減できた。

図 10 は、rabbit モデルに変形を施し、それによって生じた穴を提案法によって穴埋めした例である。腰の部分に生じた穴が自動的に検出され、穴埋めされている。

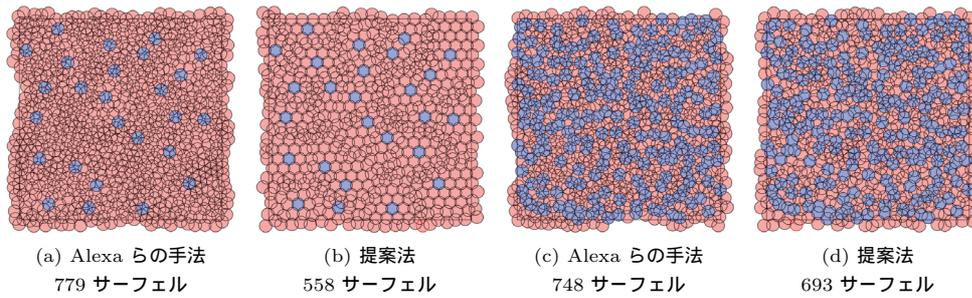


図7 提案法と Alexa ら<sup>1)</sup>の手法との比較. 青は入力されたサーフェル, ピンクは新しく追加されたサーフェルを表す. (a) と (b) は入力が少ない場合 (26 サーフェル), (c) と (d) は入力が多い場合 (196 サーフェル).

Fig. 7 Comparisons with Alexa et al.'s method<sup>1)</sup> and the present method. (a) and (b): sparse input (26 surfels), (c) and (d): dense input (196 surfels).

図 11 は提案法を用いて多重解像度のモデルを作成した例である。まず MLS 曲面によって近似した曲面に半径の大きなサーフェルを配置して穴埋めし、半径を小さくしてさらに穴埋めする、という手順を繰り返すことで、容易に多重解像度のモデルが生成できる。サーフェルを配列のようなデータ構造に逐次的に格納することにより、粗いレベルのサーフェルはより細かいレベルを構成するのにも使われる。このようなデータ構造は描画時に動的な分岐を必要としないため、GPU での描画に適している。さらに、提案法では半径を柔軟に変更することができるため、モデルの解像度をスムーズに変化させることができる。ここでは、粗いモデルのサーフェルの半径を半分にしていくことで多重解像度のモデルを作成した。解像度を制御するための最適な方法については今後検討したい。

## 5. まとめ

本稿では、サーフェルを用いて描画する際に穴が生じないように、アップサンプリングを行う新手法を提案した。提案法の特徴は以下の通りである：

- 各サーフェルの周囲が近傍のサーフェルによって覆われているかを検査することにより、穴を検出できる。
- サーフェルの半径を指定することでサンプリング密度を調整できる。
- 穴を自動的に検出し、埋めることができる。
- サーフェルの重なりをできるだけ小さくし、サーフェルの分布をできるだけ一様にする。
- 多重解像度データを生成できる。

今後の課題として、提案法は経験則に基づいており、より厳密な理論的裏付けが必要である。また、より正確な穴

の検出のために、接平面ではなく曲面を考慮することが挙げられる。精密なモデルにおける鋭いエッジ部分の扱いも必要である。

## 参考文献

- 1) M. Alexa, J. Behr, D. Cohen-Or, S. Fleishman, D. Levin, and C. T. Silva. Computing and rendering point set surfaces. In *IEEE Transactions on Visualization and Computer Graphics*, pages 3–15, Jan 2003.
- 2) G. Guennebaud, L. Barthe, and M. Paulin. Dynamic Surfel Set Refinement for High Quality Rendering. In *Computer & Graphics*, pages 827–838. Elsevier Science, 2004.
- 3) G. Guennebaud, L. Barthe, and M. Paulin. Interpolatory refinement for real-time processing of point-based geometry. In *Proceedings of EUROGRAPHICS 2005*, pages 657–666, August 2005.
- 4) D. Levin. Mesh-independent surface interpolation. In *Geometric Modeling for Scientific Visualization*, pages 37–49, 2003.
- 5) A. Okabe, B. Boots, K. Sugihara, and S. N. Chiu. *Spatial Tessellations: Concepts and Applications of Voronoi Diagrams*. Wiley, 1992.
- 6) H. Pfister, M. Zwicker, J. v. Baar, and M. Gross. Surfels: Surface elements as rendering primitives. In *Proceedings of ACM SIGGRAPH 2000*, pages 335–342, July 2000.
- 7) C. Shen, J. F. O'Brien, and J. R. Shewchuk. Interpolating and approximating implicit surfaces from polygon soup. In *Proceedings of ACM SIGGRAPH 2004*, pages 896–904, August 2004.
- 8) K. Shimada and D. C. Gossard. Bubble mesh: automated triangular meshing of non-manifold geometry by sphere packing. In *SMA '95: Proceedings of the third ACM symposium on Solid modeling and applications*, pages 409–419, 1995.
- 9) M. Stamminger and G. Drettakis. Interactive sampling and rendering for complex and procedural geometry. In *Rendering Techniques 2001: 12th Eurographics Workshop on Rendering*, pages 151–162, June 2001.
- 10) G. Turk. Re-tiling polygonal surfaces. In *Computer Graphics (Proceedings of SIGGRAPH 92)*, pages 55–64, July 1992.

(2007年5月27日受付)

金森由博



2004 年, 東京大学理学部卒。2006 年, 同大学院情報理工学系研究科修士課程修了。現在, 同大学院情報理工学系研究科博士後期課程在学中。コンピュータグラフィックスの研究に従事。情報処理学会会員。

高橋成雄



1997 年東京大学理学系研究科情報科学専攻博士課程修了。博士(理学)。群馬大学工学部助手。同大学総合情報処理センター助教授, 東京大学大学院総合文化研究科助教授を経て, 現在, 東京大学大学院新領域創成科学研究科複雑理工学専攻准教授。ビジュアライゼーション, 視覚応用システム, 幾何形状モデリング, 地理情報システムなどに興味を持つ。電子情報通信学会和文論文誌(A)編集委員 IEEE CS, ACM, Eurographics, 情報処理学会, 可視化情報学会, 日本視覚学会各会員。

西田友是(正会員)



1971 年, 広島大学工学部卒業。1973 年, 同大学院工学研究科修了。同年, マツダ(株)に入社。1979 年, 福山大学工学部講師。1984 年, 同助教授。1990 年, 同教授。1998 年, 東京大学理学部教授。1999 年, 同大学院新領域創成科学研究科教授となり, 現在に至る。2005 年, ACM SIGGRAPH より Steven A. Coons 賞を受賞。コンピュータグラフィックスの研究に従事。工学博士。情報処理学会, 電子情報通信学会, 画像電子学会, ACM, IEEE 各会員。

画像電子学会, ACM, IEEE 各会員。

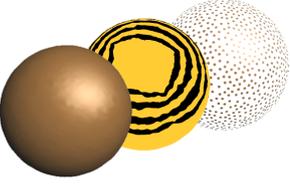
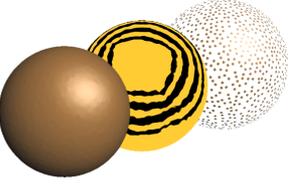
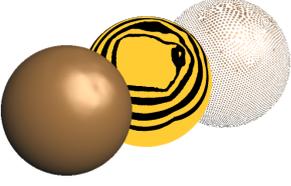
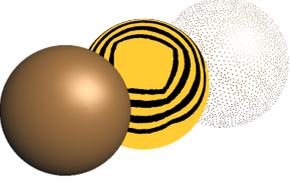
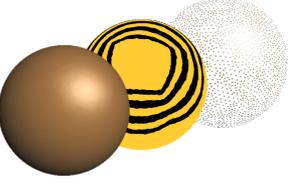
入力	Guennebaud らの手法 <sup>3)</sup>	Alexa らの手法 <sup>1)</sup>	提案法
 sphere 154 サーフエル	 (a) 細分割 4 回, 平均半径 0.018 4,182 サーフエル, 0.74 秒	 (b) 半径 0.018 2,224 サーフエル, 0.41 秒	 (c) 半径 0.018 2,100 サーフエル, 0.09 秒
	 (d) 細分割 5 回, 平均半径 0.010 12,512 サーフエル, 2.00 秒	 (e) 半径 0.010 7,188 サーフエル, 0.97 秒	 (f) 半径 0.010 6,127 サーフエル, 0.27 秒
 maxplanck 5,813 サーフエル	 (h) 細分割 5 回, 平均半径 0.0021 296,116 サーフエル, 68.50 秒	 (i) 半径 0.0021 138,217 サーフエル, 21.31 秒	 (j) 半径 0.0021 132,415 サーフエル, 7.25 秒
 igea 134,344 サーフエル	 (k) 細分割 2 回, 平均半径 0.0012 1,057,633 サーフエル, 60.33 秒	 (l) 半径 0.0012 446,328 サーフエル, 63.78 秒	 (m) 半径 0.0012 419,779 サーフエル, 22.57 秒

図 8 既存手法との比較.

Fig. 8 Comparisons with the previous methods.



図 9 曲率に応じて半径を変えて穴埋めした結果. サーフェルの数は (a)45,976(曲率が大きくなるにしたがって青から赤を変えている), (b)67,568, (c)116,390 となっている.

Fig. 9 Filling holes with surfel radii dependent on surface curvatures. The numbers of surfels are (a) 45,976 (colored according to surface curvatures which increase from blue to red), (b) 67,568 and (c) 116,390.

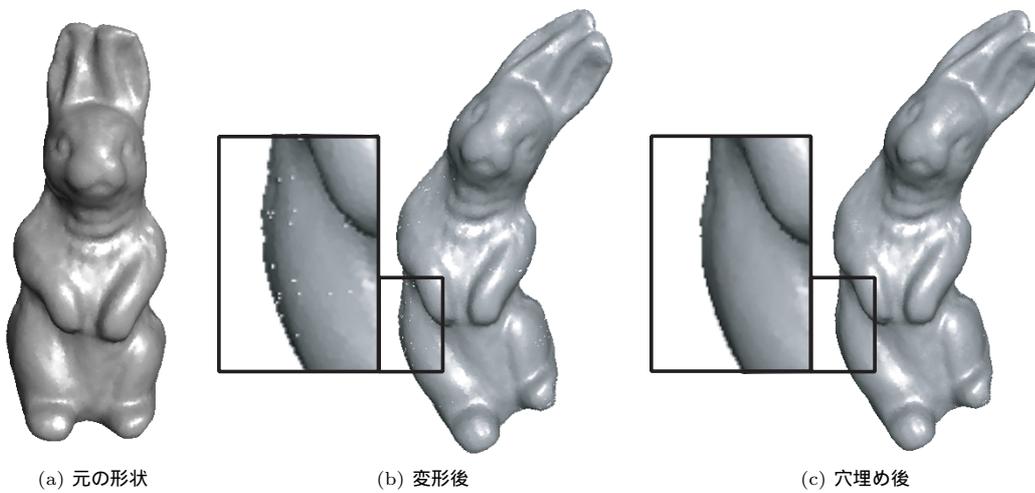


図 10 変形された形状の穴埋め. サーフェル数は穴埋めによって 23,659 から 28,413 に増加した.

Fig. 10 Hole-filling of a deformed object. The number of surfels increased from 23,659 to 28,413.

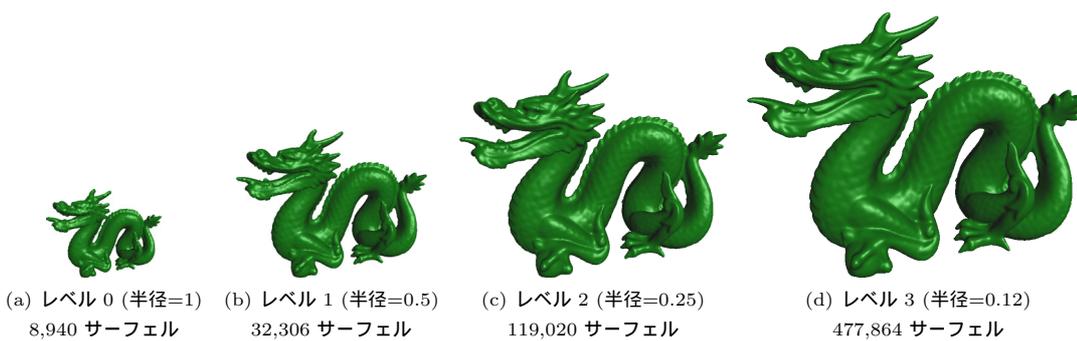


図 11 多重解像度の Stanford dragon モデル.  
Fig. 11 The level-of-detail data for the Stanford dragon.