I-029

点群データを用いた点描レンダリング
Interactive Stipple Rendering for Point-Cloud

粟野 直之† 西尾 孝治† 小堀 研一†
Naoyuki Awano Koji Nishio Ken-ichi Kobori

1. はじめに

3 次元形状のノンフォトリアリスティックレンダリング（NPR）は，従来メッシュデータに対して行われてきた．これは，メッシュデータの位相情報を利用することで強調したい部分の抽出が容易なためであると考えられる．しかし，近年ポイントベースレンダリングの研究が盛んに行われるようになり，3 次元スキャナから取得できる点群データを用いた NPR についても研究されるようになってきた[1][2]．そこで本研究では点群データの表示に適していると考えられる点描を表現する NPR 手法を提案する．提案手法では，一般にレンダリング時に必要とされる法線ベクトルを用いずイメージベースでのシェーディングを行い，描画する点の数を段階的に調節できるようにすることで複数の階調を表現できるようにする．また，実時間での視点変更も可能にする．

2．点描レンダリング

2.1 手法の概要

提案手法では法線ベクトルを持たない点群データを入力とし，2 種類の処理に大別することができる．最初に，陰点消去として本来形状の表面が存在することで隠される点を不可視とする．次に，点描は点の密度を変化させることで陰を表現するため，陰点消去後に残された可視点の一部を非表示とすることでシェーディングを表現する．

2.2 陰点消去

メッシュデータにおける陰面消去の代表的な手法として，スクリーンを構成するピクセル毎に奥行き判定を行い，最も手前にある面情報のみ描画する Z バッファ法がある．提案手法では Z バッファ法をもとにした手法を用いて陰点消去を行う．

まず，スクリーン位置にスクリーン解像度と同等かそれ以上のテクスチャを用意し，点群データの各点をテクスチャの対応するピクセルに格納する．このとき，各ピクセルには最も Depth 値が小さい点とその Depth 値を格納する．しかし，点群データには明確な形状表面の定義がないため，各点間の隙間から形状表面によって隠されるべき点が可視となる．

そこで，図 1(a)に示すように点 A・B・C をピクセルに格納する際，同図(b)に示すように近隣のピクセルにも各点を格納しておくことで各点間の隙間を埋める．ここで，同図における各点の Depth 値は A < B < C としている．すべての点をテクスチャに格納した後，テクスチャに残っている点をすべて可視とすることで陰点消去を行う．

しかし，この方法では点が密集している個所において多くの点が不可視となるため，全体的に密度が均一になるように消去されることから薄い印象を与えてしまう．

そこで，格納される Depth 値を段階的に変化させて格納することで，点が密集している個所においても多くの点を可視とする．具体的には，図 1 における Depth 値が図 2(a)に示すような値で格納された場合，例として同図(b)に示すように同図中の○で示す中央のピクセルから遠ざかるほど Depth 値を大きくして格納する．ここで，各ピクセルには最も小さい Depth 値を格納する．

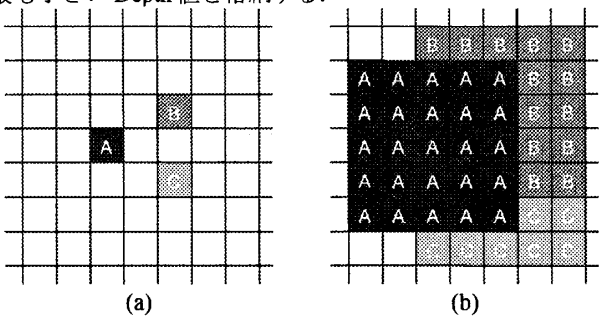


図 1 ピクセルへの格納

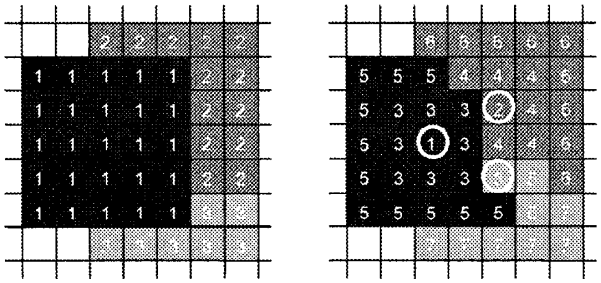


図 2 Depth 値の操作

2.3 シェーディング

本来，メッシュデータにおけるシェーディングは法線ベクトルと光源ベクトルを用いて計算される．提案手法では法線ベクトルを入力とせず，イメージベースでのシェーディングを行う．その方法として，図 3 に示すように点群データをスクリーン平面に投影した際に，スクリーン平面上において形状のシルエット付近に点が多くなる特徴を利用する．提案手法ではスクリーン平面に投影された点を対象にシェーディングを行うため，光源は視点位置に設置し，平行光源とする．

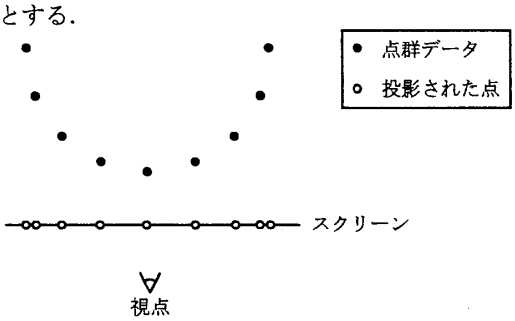


図 3 スクリーンへの投影

† 大阪工業大学

まず，スクリーン解像度と同等かそれ以上のテクスチャを用意し，1 ピクセルを 1 つのリーフノードとする完全 4 分木を作成しておく．図 4 では用意するテクスチャを 4×4 とし，分割 Level 2 の完全 4 分木を作成した例を示している．そして，各ピクセルには投影される点の中で最も小さい Depth 値を持つ 1 点をそれぞれ保持しておく．

次に，前述したように点の一部を非表示とすることでシェーディングを行う．具体的には，完全 4 分木のある Level を指定し，指定した Level のノード内に保持されている点をすべて非表示とする．例として，図 4 において Level 1 を指定すると，対応する 4 ノード内の○で示す 4 点が非表示となる．

以上の処理によって全体的に均等に点を非表示とするため，シルエット付近において多くの点を残したままシェーディングを行うことができる．

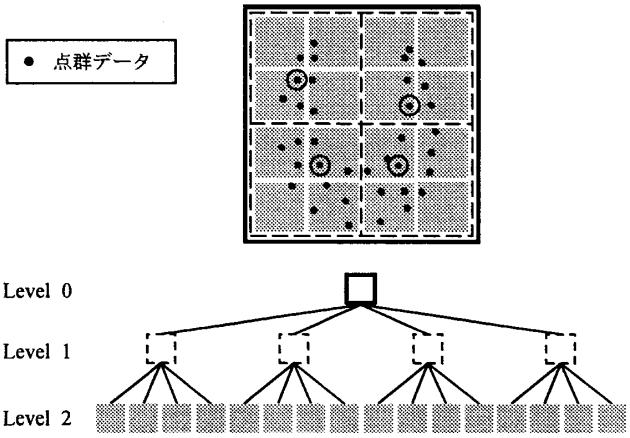


図 4　完全 4 分木

3. 実験結果

提案手法の有効性を検証するために実験を行った．実験結果を図 5・6 に示す．[ ]内の数値は入力点群データの点数を表している．

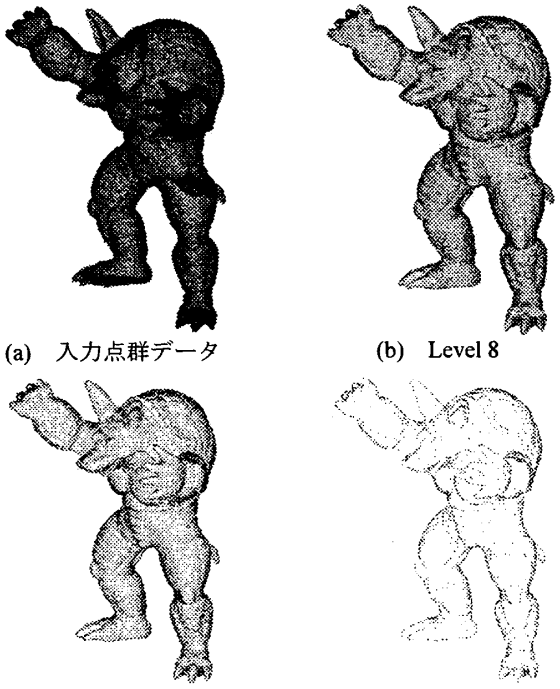
まず，図 5 に示すように表示する点数を段階的に調節できていることがわかる．ここで，同図の Level は非表示の指定 Level を表している．また図 6 に示すように，陰点消去時における Depth 値の操作によって実行結果が明瞭になっていることがわかる．従って，Depth 値を操作することでシルエットの濃さを変化させることができる．

次に，陰点消去とシェーディングを行う処理時間の合計を計測した．計測環境は Core2Duo T7200 2.0GHz，2.0GB RAM とし，テクスチャ解像度を 2048×2048 とした．処理時間を表 1 に示す．同表より，高速にレンダリングできていることがわかり，実時間での視点変更が可能であることがわかる．

4. おわりに

本研究では点群データの表示に適していると考えられる点描レンダリングの NPR 手法を提案した．提案手法では法線ベクトルを持たない点群データを入力とし，イメージベースでの陰点消去とシェーディングを行った．また，シェーディングの度合いを調節できるようにした．実験より，陰点消去とシェーディングが行えることを確認し，実時間での視点変更が可能な処理時間であることを確認した．

今後の課題として，現在では平行光源を用いたシェーディングであるが，全体に拡散する光を表現していると考えられることから，鏡面反射におけるハイライトのような部分的に強調されたシェーディングを行えるようにすることが考えられる．



(a)　入力点群データ　(b)　Level 8

(c)　Level 9　(d)　Level 10

図 5　実行結果 armadillo [172,974]

(a)　Depth 値の操作なし　(b)　Depth 値の操作あり

図 6　実行結果 lucy [262,909]

表 1　処理時間

形状名	点数	処理時間（msec）
armadillo	172,974	69
lucy	262,909	73
dragon	437,645	96

参考文献

[1] A.Runions, F.Samavati, P.Prusinkiewicz, “Ribbons”, Computer Graphics International (2007)

[2] H.Xu, M.X.Nguyen, X.Yuan, B.Chen, “Interactive Silhouette Rendering for Point-Based Models”, Eurographics Symposium on Point-Based Graphics (2004)