# 点群データからの三角形メッシュの生成

辻森　裕樹　　西尾　孝治　　小堀　研一
大阪工業大学

## 1. はじめに

近年，3 次元スキャナの発達により点群と呼ばれる 3 次元形状の表面を構成する高密度な測定データが得られるようになってきている．点群は，位相情報をもたないため，CG の分野ではサーフェスモデルに変換されることが多い．しかし，点群がノイズを含んでいる場合や点群の一部が欠損している場合，点の密度が均一でない場合は形状の表面を決定することが難しいといった理由から表面を構成する穴のないメッシュの作成は難しい問題となっている．点群をサーフェスモデルに変換する従来研究に，点の法線情報を用いずに隙間のない 3 次元形状を生成する Alexander らの手法[1]が知られている．この手法では，点群をボリュームデータに変換し，形状の表面と考えられる位置にグラフを作成する．そして，グラフの最少カットを求めることによって最も表面に近い部分を推定しサーフェスモデルへの変換を行っている．しかし，階層的な処理を行っているため細かな解像度を要求すると処理時間が増加するという問題がある．

そこで本研究では，この手法のグラフの構成方法を改良してグラフのエッジ数を減少させることにより，点群からサーフェスモデルへの変換速度を向上させる．

## 2. 従来法

従来法の入力は，法線情報を持たない点群である．従来法の処理は，粗い解像度から開始し，段階的に解像度を細かくすることで表面に近い位置を推定する．図 1 に従来法の処理の流れを示す．まず，点群をボクセルに変換する．そして，6 近傍に膨張処理を適用して形状の概形を作成する．次に，形状の概形を構成する各ボクセルにグラフを作成する．作成したグラフの最大フローを計算し，最小カットを求める．この最小カットが内部に存在するボクセルを形状の表面に近い位置にある形状表面ボクセルとして決定する．より細かな解像度を必要とする場合には，ボクセル空間の解像度を細かく変換し，先ほど決定した形状表面ボクセルと入力の点群が存在する位置を形状が存在する初期ボクセルとする．そして，膨張処理から先程と同様に処理を繰り返し形状の表面に近いボクセルを新たに推定する．必要な解像度に達した場合は，形状表面ボクセルに対して三角形メッシュの生成を行う．最小カットに対応する形状表面ボクセルの辺が隣接する形状表面ボクセルと同様の辺を必ず共有することを利用し，各々のボクセルの中心を結ぶことで三角形の辺を作成し三角形メッシュの生成を行う．

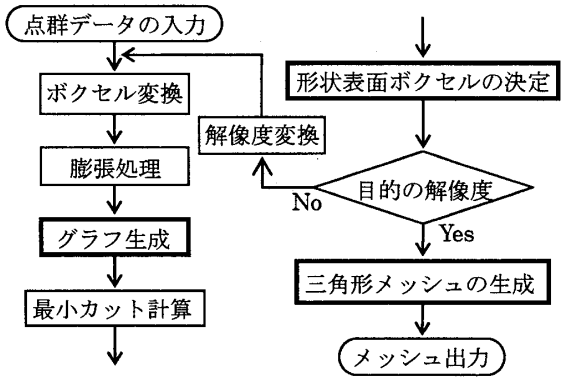
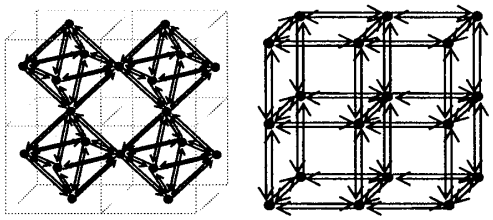



図 1　従来法の処理の流れ

## 3. 提案手法

提案手法では，従来法のグラフの構成方法を変更してグラフのエッジを減らし，最小カット計算時の処理コストを下げる．従来法では，図 2(a)に示すように形状の概形を構成する各ボクセルの内部に内接する正八面体の辺を使用しグラフを構成する．これに対して，本研究では，同図(b)に示すようにグラフの構成に立方体の辺を用いる．エッジを共有するように配置するため，エッジ数を削減させることができる．



(a)従来法　　(b)提案手法

図 2　グラフの違い

Mesh Reconstruction from Point Cloud
Hiroki Tsujimori,Koji Nishio and Ken-ichi Kobori
Osaka Institute of Technology

### 3.1 グラフの生成

図1のグラフ生成を行う際に設定する各エッジの重みは，エッジがボクセルの内部に存在することを利用する．あらかじめ各ボクセルで入力の点群からどれだけ離れているかを計算しておき，この値を重みとする．しかし，提案手法では，グラフのエッジがボクセル内部に存在しないため，従来法のように重みを設定することが出来ない．そこで，提案手法では，エッジに隣接するボクセルに設定された重みの平均を各エッジの重みとしグラフを生成する．

### 3.2 形状表面ボクセルの決定

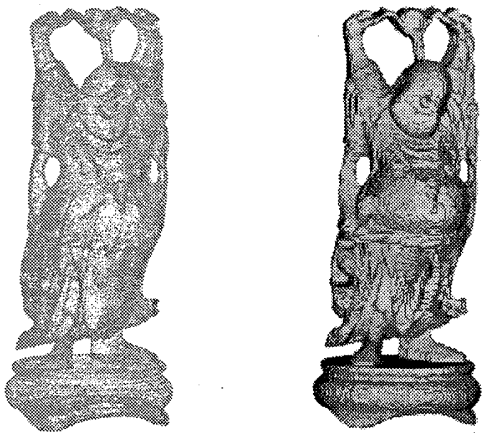
図1の解像度の変換では，新たに細かくした解像度で形状表面が存在する位置を決定するために形状表面ボクセルを利用する．しかし，提案手法では最小カットがボクセルの内部に存在しないため処理対象とする形状表面ボクセルを一意に決定することができない．そこで，提案手法では，最小カットに隣接するボクセル全てを形状表面ボクセルとし，階層的な処理に利用する．

### 3.3 三角形メッシュの生成

図1の三角形メッシュの生成では，形状表面ボクセルと最小カットに対応するボクセルの辺を利用して生成される．しかし，提案手法では，最小カットが通る辺しか出力されないため従来のメッシュ生成法は適用できない．そこで，最少カットを求めたグラフの各ノードがSourceかSinkのどちら側に属しているかといった情報を利用してメッシュ生成を行う．Sourceと判断されたノードは形状の外側，Sinkと判断されたノードは形状の内側に位置する．このことから，SourceとSinkといった2値情報を利用するとマーチンキューブス法と同様の三角形パッチパターンでメッシュ生成を行うことができる[2]．提案手法では，最小カットと判断されたエッジの中点に三角形の頂点を生成し，マーチンキューブス法の三角形パッチパターンに従ってメッシュを生成する．

## 4. 実験

提案手法を用いて，メッシュ生成の実験を行った．実験環境は，CPU：Pentium4 3.2GHz，Memory：2GBである．図3にメッシュ生成結果の一例，表1に処理時間，表2にグラフのエッジ数とメッシュ数を示す．なお，処理時間には点群データの読み込み時間を含んでいない．解像度は256×256×256である．



(a) 入力点群　　(b) メッシュ

図3　buddhaのメッシュ生成結果

表1　処理時間(s)

| | 従来法 | 提案手法 |
|---|---|---|
| buddha | 5.56 | 2.73 |
| dragon | 7.13 | 3.78 |
| armadillo | 7.70 | 4.31 |

表2　グラフのエッジ数とメッシュ数

| | グラフのエッジ数 | | メッシュ数 | |
|---|---|---|---|---|
| | 従来法 | 提案手法 | 従来法 | 提案手法 |
| buddha | 7,399,416 | 2,343,930 | 149,000 | 250,160 |
| dragon | 9,292,644 | 3,094,636 | 181,872 | 326,198 |
| armadillo | 8,906,292 | 2,985,896 | 177,366 | 312,540 |

表1と表2より，エッジ数を削減したことによって処理時間が4割程度改善できていることがわかる．しかし，提案手法ではメッシュ数が増加している．これは，マーチンキューブス法を利用してメッシュ生成を行ったからである．

## 5. おわりに

グラフの構成を変更することによって，グラフのエッジ数を削減し，処理時間を改善する手法を提案した．実験により，処理時間が改善されていることを確認した．今後の課題として，メッシュ数の削減が挙げられる．

### 参考文献

[1]Alexander Hornung, Leif Kobbelt : "Robust Reconstruction of Watertight 3DModels from Non-uniformly Sampled Point Clouds Without Normal Information", Eurographics Symposium on Geometry Processing 2006, 2006

[2]Willian E.Lorensen and Harvey E.Cline : Marching Cubes "A high resolution 3D surface constructionalgorithm", Proc.of SIGGRAPH1987, pp.163－169, 1987