大規模点群データの平滑化手法に関する研究（第 2 報）*

－大規模点群平滑化のためのストリーミング処理－

増田　宏**　　村上　健治***

A Study on Smooth Surface Reconstruction from Large-Scale Noisy Point-Clouds (2nd report)
- Streaming Processing for Smoothing Large-Scale Point-Clouds -

Hiroshi MASUDA and Kenji MURAKAMI

The state-of-the-art phase-based scanner is suitable for acquiring 3D shapes of large facilities, because it can acquire hundreds of millions point data in several minutes. Since acquired data are very noisy, smoothing operations are very important. However, point data from large facilities are too large to process in the memory of common personal computers. This paper proposes a new streaming smoothing operator for very large-scale point-cloud acquired by phase-based or time-of-flight scanners. Our method reads point data on a hard disk and sequentially processes them in a small region on RAM. This method is based on the fact that point data from a laser scanner can be converted into a height-field on a sphere. Our experimental results show that our method could successfully produce a smooth mesh model from large-scale point data with the limited memory size.

Key words: geometric modeling, point-cloud, smoothing, streaming, reverse engineering, as-built modeling

1. 緒　　言

老朽化した生産設備やプラント施設の改修やメンテナンスを迅速に行うには，点群計測に基づく現物の 3D モデリングが有効である．そのための計測装置としては，位相差方式のレーザスキャナが適していると考えられる．この装置は膨大な点群を短時間で計測できるという特徴を持つため，計測時間が限られている生産設備やプラント施設の計測に都合がよい．本研究で用いた位相差方式のレーザスキャナ（Z+F 社 Imager5003）では，4 分弱で 5000 万点，7 分弱で 2 億点の計測が可能である [1)]．

しかしながら，位相差方式の点群計測装置では計測点のノイズが大きく，また，大量の異常値が含まれる．さらに，一度の計測で取得される点の個数が膨大である．大型設備において，数十 m 離れた小型の部材をモデリングするには，充分な点群密度で計測しなければならず，大容量の点群処理が不可欠となる．

位相差方式のレーザスキャナで取得された点群のノイズ処理に関しては，第 1 報において，ロバスト推定を用いた平滑化手法を提案し，異常値が多い点群でも安定して平滑化が行える手法を示した [2)]．ただし，この方法を大規模点群に対して適用しようとすると，安価な 32 ビット計算機では RAM が最大 4GB という制約もあり，使用するメモリ量に対して充分な配慮が必要となる．さらに，レーザスキャナの性能が年々向上し，単位時間当たりの計測点群数が飛躍的に伸びていることから，さらなる大容量化も考えておく必要がある．また，近年では生産拠点がグローバルに展開され，改修を必要とする生産設備が海外の新興国にあることも多くなっていることから，メモリ制約の強い軽量のノート PC を用いて現場で点群処理できることも求められている．こうしたことから，本研究では，少ないメモリでも大量の点群を処理できる手法について考える．

一般に，計算機においては RAM の容量に比べて，ハードディスクは遥かに大きい容量を持つ．そのため，データ量が RAM の容量を超えたときには，ハードディスク上に仮想メモリを設定して処理することがよく行われる．しかし，ハードディスクへのアクセス速度は RAM に比べると非常に遅い．特に，ハードディスク上のデータをランダムにアクセスしようとすると，処理速度は極端に遅くなる．そのため，仮想記憶を用いた場合，大量点群を効率的に処理することは困難である．

ハードディスクでは，連続したデータの読み込みはランダムアクセスに比べて非常に高速に行える．この性質を用いて大量のデータを効率的に処理する方法として，ストリーミング処理が知られている．この方法では，ハードディスクからデータを読み込んだ順番にデータ処理を行い，その結果を順次出力していく．処理が完了して不要となったデータはメモリから削除していくため，限られたメモリ空間でデータ処理ができる．また，ハードディスクへのアクセスはデータを順番に読み込んでいくことだけなので，大量のデータをハードディスクから読み込んで，すべてを RAM 上に展開して処理した場合とほぼ同等の時間で処理が行える．

ただし，ストリーミング処理を行うためには，ハードディスク上のデータが，データ処理に適した順番に並んでいることが不可欠である．たとえばビデオデータでは，画像データが時系列的に整列しているため，呼んだ順番に画面に表示するストリーミング処理が容易である．同様の考え方を点群の平滑化に適用するためには，平滑化に必要となる近傍点のデータが，ハードディスク上においても近傍にあることが必要である．

本研究では，点群から平滑化されたメッシュモデルを作成するためのストリーミング処理を実現する．まず，座標を球面座標 (θ,ϕ,r) で表現することで，ストリーミング処理に適した形式であるハイトフィールドとして点群データを処理できることを示す．次に，平滑化のための近傍データのバッファリングとメモリ解放のタイミングを適切に管理することで，大規模点群の平滑化がストリーミング形式で行えることを示す．

以下，第 2 章では，位相差方式の点群計測装置から出力された点群ファイルを平面上に写像する方法を示す．第 3 章では，ス

*　原稿受付 平成 21 年 4 月 30 日
**　正 会 員正 会 員 東京大学大学院（東京都文京区本郷 7-3-1）
***　東京大学大学院（同上）

トリーミング方式でメッシュを生成する既存手法を説明する．第 4 章では，平滑化のためのストリーミング処理手法について示し，第 5 章で評価実験を示す．最後に結論を述べる．

2. 球面座標を用いた 2 次元平面への写像

点群計測装置から出力される点群の並び順は，計測方法に依存する．代表的な点群計測方式には，三角測量方式，位相差方式，飛行伝播時間計測方式の三つがあるが，比較的大規模な対象物の計測では，位相差方式と飛行伝播時間計測方式が用いられる．これら二つの方式の計測装置では，光源からレーザ光を照射して，その反射波の飛行時間から距離を計算している．レーザ光の照射角度は，**図 1** に示すように，レーザ発生器を回転させて制御する．まず，角度 θ を固定して ϕ の向きに 360 度回転させる．それにより，前方と後方の点に関して距離が計測される．ϕ 方向に一周したら，θ を微小角度ずつ 180 度まで回転させて同様の計測を行うことで，全周囲の点群を取得する．

このように計測される点群の座標を，**図 2** に示す球面座標系を用いて表す．この座標系では，レーザ光の光源が原点となり，レーザ光の照射方向が二つの角度 (θ,ϕ) によって表現される．また，原点からの距離を r とし，3 次元座標を球面座標 (r,θ,ϕ) によって表現する．このとき，直交座標 (x,y,z) と球面座標 (r,θ,ϕ) の関係は以下の式で表される．

$$\begin{aligned} x &= r\cos\theta\sin(-\phi) \\ y &= r\sin\theta\sin(-\phi) \\ z &= r\cos\phi \end{aligned} \qquad (1)$$

なお，我々が使用した計測装置の仕様では，有効な角度の範囲は，$0 \le \theta \le 180$，$-155 \le \phi \le 155$ （degree）である [1]．

計測された点群を $\theta-\phi$ 平面上にプロットすると，**図 3** に示すように，ϕ 軸，θ 軸に沿ってほぼ等間隔で整列する．この画像の明暗値は，レーザの反射光の強さ I を $0 \le I \le 255$ の整数値に変換して表示したものである．点群は，計測された順番に格納されており，ファイル中では図 3 の矢印で示した向きに，左から右へと格納されていく．この例では計測点は約 5000 万点であり，ϕ 方向に約 10000 点，θ 方向に約 5000 点である．

計測された点群は $\theta-\phi$ 平面上で重なることなく分布し，3 次元空間で近傍の点は $\theta-\phi$ 平面上でも近傍点となる．そのため，距離 r を高さと見做すならば，計測された点群を $\theta-\phi$ 平面におけるハイトフィールドと考えることができる．

式(1) によって直交座標 (x,y,z) と球面座標 (r,θ,ϕ) は相互に可逆な変換が可能である．この性質を利用して，大規模点群の平滑化のための計算ではこれらの表現を組み合わせ，近傍点探索に関する処理は球面座標系のハイトフィールドで，また平滑化のための曲面当てはめの計算は直交座標系で行うことを考える．

3. 大規模ハイトフィールドのメッシュ分割

3.1 ハイトフィールドのメッシュ分割

位相差方式と飛行伝播時間計測方式の点群計測装置では，球面座標を用いて，ハイトフィールドに変換できる．点の並び方に規則性のあるハイトフィールドに関しては，メモリ容量を超えるデータをストリーミング形式でメッシュ分割できる方法が知られている．本研究では，Isenburg らの提案したストリーミングメッシュ [3)4)] を応用することで，大規模点群の平滑化をストリーミング形式で実現することを考える．

なお，図 3 に示すように点群は規則正しく並んでいるので，この並び順だけを用いてメッシュ生成を行うことも可能である．しかし，計測された点には所々に抜けが生じ，また，点群の平滑化によって点の位置が微小距離ずれるので，点の並び順だけに頼ったメッシュ生成では，面の裏返りが生じ得る．そのため，本研究では，ストリーミングメッシュに基づくメッシュ分割法を用いている．以下において手法の概要を説明する．

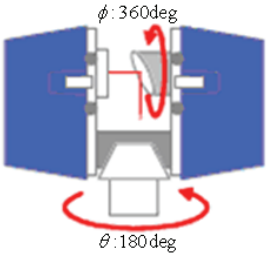


Fig.1 Measurement by phase-based scanner

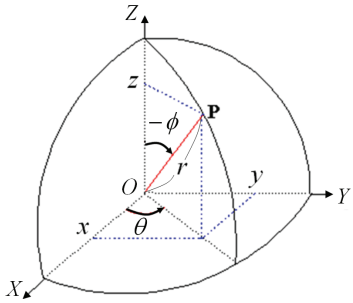


Fig.2 Spherical coordinate system

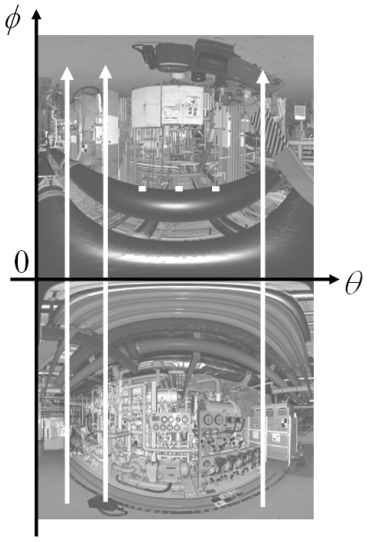


Fig.3 Measured data and the order of points

3.1 ストリーミングメッシュ

メッシュデータの一般的な記述法として，VRML, OBJ, PLY などのメッシュフォーマットが広く用いられている．これらの表現では，**図 4**(a) のように最初にすべての座標が羅列して記述され，次に面のデータが記述される．頂点には記述された順番に頂点番号が付与され，面を構成する頂点列は頂点番号で指定される．

ただしこの表現では，面を作成するために，すべての頂点座標を一旦メモリ上に保持することが必要である．そのため，大規模なメッシュデータの処理には膨大なメモリ空間を必要とする．

ストリーミングメッシュは，RAM の容量を超える大規模メッシュを扱うためのフォーマットである [3)]．図 4(b) にその形式を示す．この例では，まず 3 個の頂点 v_1, v_2, v_3 の座標が記述され，そ

の直後に3頂点を用いた三角形 f_1 のデータが頂点番号を用いて記述されている．また，三角形 f_3 においては，頂点番号として負の値を用いることで，頂点1がそれ以降は利用されないことを示している．ここでは，負の符号を終了タグと呼び，頂点データのメモリ解放のタイミングを示すために用いられる．

このような形式で記述することにより，頂点と面を適切な順序でファイルに書き込んでおけば，少ないメモリでもファイルから読み込んだ順番にメッシュ再構成することが可能となる．

3.2 ストリーミングメッシュの生成法

次に，ハイトフィールドで記述された点群からストリーミングメッシュを生成する方法について説明する．本研究では，第2章で示したように，本研究で扱う点群はハイトフィールドに変換することが可能なので，この方法を適用することができる．

点群からのメッシュ生成には，逐次頂点挿入法が用いられることが多い．この処理では，**図5**に示すように頂点を一個ずつ挿入しながら三角形メッシュを成長させていく．頂点を挿入する際には，まずその頂点が含まれる三角形を探す（図 5(a)）．次に，その三角形と隣接する3個の三角形を消去する（図5(b)）．最後に，消去された三角形の頂点を用いて，三角形を再構成する（図5(c)）．すべての頂点を挿入したら三角形メッシュ分割が完了する．

この処理をストリーミング形式で行う手法として，Isenburg らは以下の手順を提案した[4]．

(1) 全体を**図6**に示すように矩形のセルに分割し，各セルの内部に含まれる点の個数を数える．そのために，一度目のファイル読み込みを行う．

(2) 2度目のファイル読み込みでは，逐次頂点挿入法を用いて三角形分割を行う．その過程で，セル内のすべての頂点の挿入が完了すれば，そのセルに終了タグを付ける．図6では，セル1-5に終了タグが付いているものとする．

(3) 終了タグの付いたセルの内部にある三角形（セル5に終了タグが付いた直後では，図6の色付きの三角形）には頂点がこれ以上挿入されることはないので，終了タグを付ける．また，終了タグの付いた三角形に囲まれた頂点(たとえば，図 6 では白丸で示した頂点）は，これ以後，頂点番号が参照されない．そこで，確定した三角形と頂点を必要に応じて終了タグを付加してファイルに書き出し，メモリから削除する．

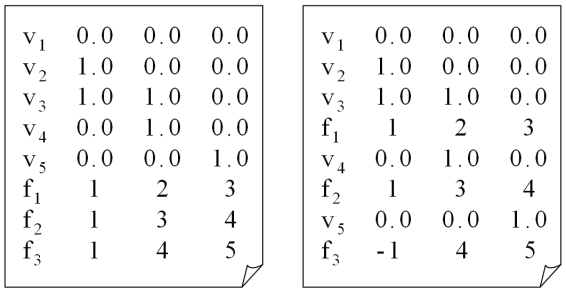
このメッシュ生成法では，終了タグのついていない頂点と面に関するデータのみを保持しておけばよく，それ以外のデータはファイルに書き出した後にメモリから削除できる．そのため，元の点群データが2次元の矩形領域に沿って充分に整列しているのであれば，大規模点群であっても少ないメモリで処理できる．

4. 大規模点群の平滑化処理

4.1 平滑化処理のための近傍点探索

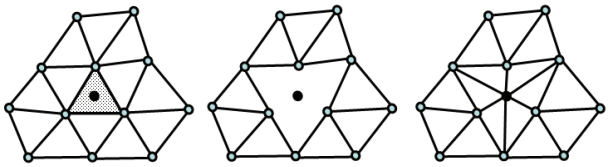
位相差方式のレーザスキャナで取得された点群のノイズ処理に関しては，第 1 報において，ロバスト推定を用いた平滑化手法を提案した[2]．我々の方法では，点群を平滑化するために，**図7**に示すように，平滑化したい点の近傍点を取得した後に2次多項式曲面を当てはめ，元の点を曲面上に投影する．その際，極端に値のずれた異常値の影響を受けにくくなるように，最適化の目的関数を工夫する．そうした目的関数として，ノイズがローレンツ分布に従うと仮定した最尤推定と，Tukey のバイウェイト推定法に基づくロバスト推定の二つの手法を提案し，計算効率の点から後者が有利であることを示した．

この方法では，図7に示したように，点 p_i を平滑化するために



(a) Traditional format (b) Streaming mesh

Fig.4 Types of mesh formats



(a) Vertex insertion (b) Deletion (c) Triangulation

Fig.5 Delaunay triangulation by vertex insertion

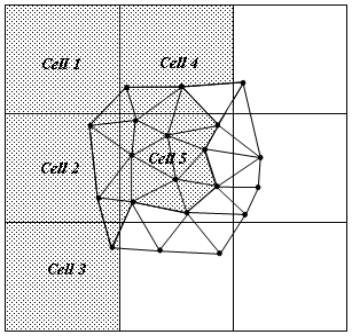


Fig.6 Triangle mesh on a segmented height field

近傍の点を取得する必要がある．ここでは，ユークリッド距離が点 p_i から近い順に k 個の近傍点を得るものとする．また，それらの点を k 近傍頂点と呼ぶ．

なお，図2, 3では $\phi = 0$ が特異点となるため，その付近の点には例外処理が必要である．特異点は球面座標の取り方に依存するものなので，$\phi = 0$ 付近の点を処理するためには，図2の Z 軸の向きを90度回転させて球面座標を再定義し，極地付近の点をその座標系で処理すればよい．ただし，本論文では，以後の議論においてこの例外処理については考えず，概ね $|\phi| > 15^\circ$ の場合を考えるものとする．このとき，我々の提案した平滑化手法の実験により，5000万個の点群で k を150〜200とすれば良好な平滑化が行えることがわかっている．

ここでは，$\theta - \phi$ 平面上でメッシュ分割したストリーミングメッシュを用いて平滑化処理を行う場合を考える．このとき，点 p_i の k 近傍頂点を得るためには，以下を満たしていることが必要となる．

(1) 点 p_i より前に読み込まれた k 近傍頂点がメモリに保持されている．

(2) 点 p_i の平滑化を行う時点では，点 p_i より後に読み込まれる k 近傍頂点が既にメモリ内に展開されている．

さらに，限られたメモリ容量で処理を行うためには，近傍頂点として明らかに利用されなくなった頂点は，メモリから削除されていることが必要である．

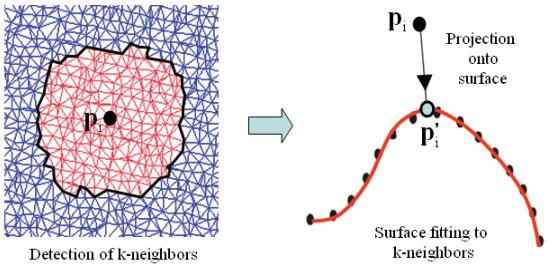


Fig.7 Surface fitting to k-neighbor vertices

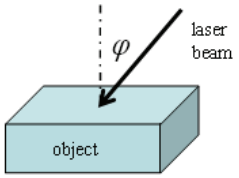


Fig.8 Angle between surface and laser beam

これらを満たすための手法を検討する．まず，k 近傍頂点を得るためには，点 p_i の前後の頂点を何個保持したらよいのかを見積もってみる．

いま，図 3 に示す $\theta-\phi$ 平面において，点群が θ, ϕ の各軸方向に等間隔で並んでいるものとする．ここで，位相距離に基づいて k 近傍頂点を得るためには，面積が k となる円を考えればよいから，メモリには少なくとも半径 $\sqrt{k/\pi}$ の円内の頂点が保持されている必要がある．このとき，点群計測装置で取得された点群が図 3 の ϕ 方向に一列あたり N 個存在するならば，平滑化する点 p_i の前後に $N\sqrt{k/\pi}$ 個の点を保持することが必要となる．

次に，k 近傍頂点を位相距離ではなく，直交座標系でのユークリッド距離で計算する場合を考える．本研究ではこちらの方法を用いる．このときには，k 近傍頂点の領域は $\theta-\phi$ 平面上で円にはならない．ここで，隣り合う 2 点間の照射角度の差を δ (radian)とすると，2 点間のユークリッド距離は図 3 の θ, ϕ 方向のそれぞれに関して，以下のように書ける．ただし，2 点の原点からの距離は共に r とし，δ は充分小さい値を取るものとする．

$$\begin{aligned} \Delta d_\phi &\approx r\delta \\ \Delta d_\theta &\approx r\delta\,|\sin\phi| \end{aligned} \tag{2}$$

この式において，$|\sin\phi|$ の項は，図 2 の球の極（$\phi=0$）に近づくほど，図 3 に示すように画像の縮尺が大きくなるのを補正する働きをする．また，**図 8** に示すようにレーザ光の照射角度が φ のときは，隣り合う 2 点は原点からの距離が等しくならず，式(2) より，点間距離 Δd は以下の範囲を取ると考えることができる．

$$r\delta\,|\sin\phi| \le \Delta d \le \frac{r\delta}{\cos\varphi} \tag{3}$$

ここで，φ の最大値を 85 度，ϕ の最小値を 15 度とすると，Δd の最大値と最小値の比 c は以下の式を満たす．

$$c \le \frac{1}{\sin 15^\circ \cos 85^\circ} \approx 44.33 \tag{4}$$

以上の議論から，保持すべき点の個数に関して最悪のケースは，k 近傍頂点の探索範囲が長径 R，短径 R/c の楕円となるときであると考えることができ，その面積を k とするとき楕円の長径は $\sqrt{kc/\pi}$ と見積もることができる．したがって，最悪のケースにおいても k 近傍頂点を得るためには，点 p_i の前後において，以下の式で定義される Ω 個の点を保持する必要がある．

$$\Omega = N\sqrt{kc/\pi} \tag{5}$$

この式に基づいて，メモリ上に保持すべき点の個数を見積もってみる．約 5000 万点の点群において，$N=10000$，$k=150$ とすると，式(5)より，平滑化する点の前後にそれぞれ約 46 万点，計 92 万点を保持しておけばよいことがわかる．また，2 億点の点群では，$N=20000$，$k=600$ として，計 368 万点を保持すればよい．これらの点群の個数は元の点群データの 2% 程度である．必要なメモリ容量は，次節で示すデータ構造を用いた場合，それぞれ 40MB，160MB 程度と見積もることができる．これらの値は，通常使われる PC の搭載メモリに比べて充分小さい量であり，最近の計算機であれば問題なく処理が行える．

4.2 平滑化処理のためのデータ構造

ストリーミングメッシュの表現では，各頂点に連結する頂点を陽に持たない．そこで，平滑化処理においては，入力されるストリーミングメッシュから，頂点間の連結関係を陽に持った連結グラフを生成して，隣接関係を保持するものとする．

ストリーミングメッシュでは，各頂点は入力された順番によって定義される頂点番号で指定される．我々が用いたデータ構造では，入力された各頂点に関して，(1) 自分自身の頂点番号，(2) 座標 (x,y,z)，(3) 連結する頂点の個数，(4) 連結する頂点の頂点番号の配列，に関してデータをメモリ上に保持する．(1)と(2) に関しては頂点データが読み込まれる度に作成される．その際，頂点データは頂点番号をキーとするハッシュを用いて管理され，頂点番号を指定すれば頂点データのアドレスが取得できるようにしておく．また (3)と(4)に関しては面データが読み込まれる度に頂点データが更新されていく．

このデータ構造を用いた場合のデータ量を見積もってみる．一般に三角形メッシュモデルにおいては，エッジの個数は頂点数の約 3 倍である．上記のデータ構造は双方向リンクを持った連結グラフに相当するため，上記の(4)で保持される頂点番号の配列の大きさは，総計で頂点数の約 6 倍となる．このことから，n 個の頂点から構成される連結グラフを保持するためのデータ量は，座標値の浮動小数点，頂点番号の整数値をそれぞれ 4 バイトとして，$44n$ バイトと見積もれる．すなわち，ある点 p_i の前後に 50 万点，計 100 万点をメモリ上に保持するならばデータ量は 44MB となる．

次に，点 p_i の前後に Ω 個の頂点を保持するためのデータ構造を考える．平滑化処理では k 近傍頂点を用いて計算する必要があるので，点 p_i が入力されても直ちに平滑化処理を行うことはできず，近傍データが揃うまで処理の開始を待たなくてはならない．また，終了タグが発行された頂点に対して直ちに削除処理を行うことはできず，周辺の頂点から近傍として参照されなくなるまではメモリ上に保持しておかなくてはならない．このタイミングを適切に管理することが必要である．

この仕組みを実現するために，本研究では，入力されたが平滑化が行われていない点群を管理する「待機点群キュー」，終了タグが発行されて削除されるのを待っている点群を保持する「終了点群キュー」の二つを用いる．

待機点群キューは連続した頂点番号を保持するので，頂点番号の最小値と最大値のみを保持する．ストリーミングメッシュ形式のファイルから読み込んだ頂点の頂点番号が順番に待機点群キューに入力され，点の個数が Ω 個を超えたときに最小の頂点番号を持つ頂点データが取り出され，平滑化処理が開始される．

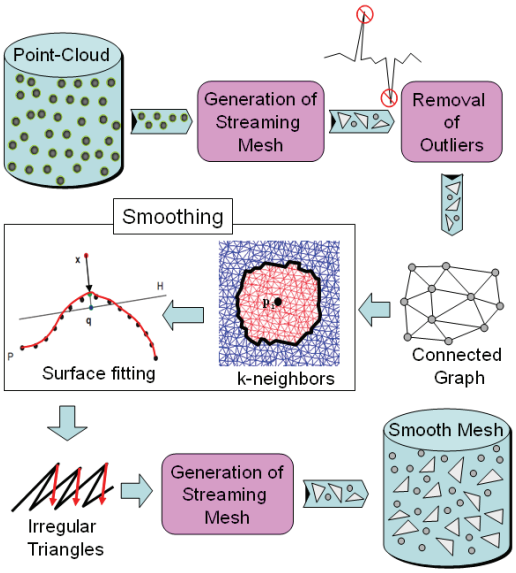


Fig.9 Pipeline for generating smooth mesh

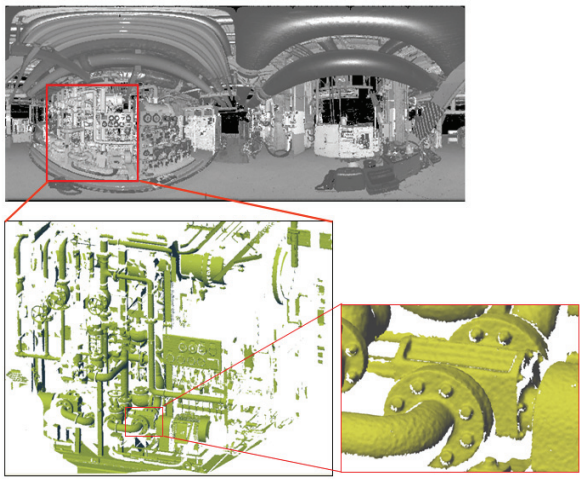


Fig.10 Smooth mesh generated from large-scale points

終了点群キューは，終了タグの発行された頂点を適切なタイミングでメモリ解放するために用いられる．ここでは，終了点群キューが持つ機能として，(1)データを追加する(put)，(2) 最も古いデータを参照する (peek)，(3) 最も古いデータを削除する(remove)，(4) 保持しているデータの個数を返す(size)，が実現されているものとする．点 p_i の平滑化が行われたとき，終了点群キューの最も古いデータの頂点番号が $i-\Omega$ より小さいならば，そのデータは頂点データのメモリから削除され，終了点群キューとハッシュテーブルからも削除される．

すべての頂点が入力された後は，待機頂点キューに残った頂点が順次処理され，平滑化処理が終了する．これらの処理を行うことによって，メモリの容量を超える大規模な点群の平滑化処理をストリーミング形式で行うことができる．

4.3 平滑化メッシュ生成のためのパイプライン処理

図 9 に大規模点群データから，平滑化されたメッシュモデルを得るまでのパイプラインを示す．まず，入力された点群がストリーミングメッシュに変換される．次に，連結頂点に比べて著しくずれている点があれば異常値として除去する．この処理は，メッシュの頂点と隣接する三角形を除去して，局所的に三角形を張りなおす操作 (vertex removal) によって実現できる．次に，4.2 で述べた平滑化処理のために連結グラフが作成され，待機頂点キューから取り出された頂点に対して k 近傍頂点が探索される．そして，それらの近傍点に対して，2 次多項式曲面を当てはめて平滑化を行う．このとき，座標の移動量が大きいと面の裏返りが発生するため，一旦，連結情報を破棄して，再度，ストリーミングメッシュを作成し直す処理を行っている．生成された平滑化メッシュは順次ハードディクスに格納される．

5. 評 価 実 験

本論文で提案した方法を用いて，大規模メッシュデータから平滑化メッシュを作成した．例題として図 3 に示した点群データを用いた．ただし，ここでは，特異点付近と計測範囲の限界付近の点を除去し，約 3898 万点をストリーミング処理した．この処理においては，まず，入力された点群をメッシュ化することで，約 7797 万個の三角形が生成された．ここから，786 万点が異常値と判定され，3112 万点から成るストリーミングメッシュを用いて点群平滑化を行った．最終的に 2777 万頂点，5247 万ポリゴンの平滑化されたメッシュモデルが作成された．

図 10 に最終的に生成された平滑化メッシュを示す．この図では，生成されたメッシュの一部を拡大して示している．

計算に要した時間は，平滑化処理に 32 分，図 9 のパイプライン全体で 37 分であった．使用した CPU は 2.66GHz の Intel Core2 Quad，メモリは 2GB である．平滑化処理では，4 個の CPU Core を用いて並列計算を行った．なお，同じ計算を Single Core と Dual Core の CPU による計算時間と比較した結果，コア数にほぼ比例して計算時間が短縮されることを確認した．このことは，GPU を用いて並列度を向上させることで計算時間が劇的に改善できる可能性があることを示唆している．

6. 結　　言

本研究では，大規模点群から平滑化されたメッシュモデルを得る処理を少ないメモリで行うための手法を示した．メッシュ生成においては，球面座標を用いることによって，ハイトフィールドにおけるストリーミングメッシュ生成の問題に帰着させられることを示した．また，平滑化のための近傍処理をストリーミング形式で行う方法を示し，大規模点群から実用的な時間で平滑なメッシュモデルが作成できることを示した．

今後の課題として，時間時間の短縮が挙げられる．点群の平滑化には非線形の最適化が必要となるため，大規模点群処理では相当な計算時間がかかってしまう．そのため，GPU を用いた実装法について検討していく予定である．

参 考 文 献

1) Technical Data IMAGER, http://www.zf-laser.com, Zoller-Fröhligh
2) 増田宏，村上健治：大規模点群データの平滑化手法に関する研究（第 1 報）－ロバスト推定に基づく平滑化手法－，精密工学会誌，掲載号未定.
3) M. Isenburg and P. Lindstrom: Streaming Meshes, Proc. of Visualization (2005) 231.
4) M. Isenburg, Y. Liu, J. Shewchuk and J. Snoeyink: Streaming Computation of Delaunay Triangulations, Proc. of SIGGRAPH (2006) 1064.