コンピュータ理工学特別研究報告書

題目　Kinect と PCL を用いた　立体物の３次元モデリング

学生証番号　０４５２４５

氏　名　山本　晃

提出日　平成 25 年 1 月 28 日

指導教員　蚊野　浩

京都産業大学

コンピュータ理工学部

要約

本研究の目的は、Kinect[1]と PCL[2]を使って、立体物の３次元モデルを作成することである。Kinect はゲーム機用のヒューマンインタフェースデバイスであるが、これを用いて立体物の形状を表す３次元点群を獲得する。PCL（Point Cloud Library）は３次元点群を処理するためのオープンソースソフトウエアライブラリである。これを用いて立体物の３次元点群を処理し、最終的に三角パッチで表現された３次元モデルを生成した。

まず、Kinect の測定精度検証を行った。白色と黒色の画用紙を壁面に貼り付け、室内照明を点灯および消灯し、500mm から 4000mm 離れた位置から画用紙面を測定した。画用紙面を平面とみなし、測定された点群とそれらにフィティングした平面との垂直距離を測定誤差として評価した。その結果、被写体との距離が近いほど誤差の標準偏差が小さく、1000mm で σ =2.5mm、2000mm で σ =7.5mm、3000mm で σ =15.0mm 程度になった。また、750mm から 3000mm の範囲であれば、画用紙の種類と照明状態によらず、誤差の標準偏差はほぼ同じであった。

次いで、PCL を用いて３次元点群を処理し、実物体の３次元モデリングが可能であることを検証した。実物体として高さ 76cm のプラスチック像を用いた。Kinect で、それを囲む 12 の位置から撮影を行い、12 個の３次元点群を獲得する。取得した点群から背景を除去し、さらに、外れ値除去、スムージング、ダウンサンプリングなどの前処理を行う。前処理した 12 個の点群を隣接する点群ごとに位置合わせ合成し、全体を表現する３次元点群に成長させる。隣接する点群を位置合わせするために、SAC-IA（Sample Consensus Initial Alignment）の初期位置合わせ後、ICP(Interative Closest Point)アルゴリズムによって高精度な位置合わせを行う。位置合わせ後、MLS（Moving Least Squares）アルゴリズムで重なり部分のダブルウォールを解消する。この手順を繰り返すことで、全体形状を表現する点群を生成できると考えた。しかし、隣接する４個程度の点群までは位置合わせできたが、それを超える点群を安定して位置合わせすることは困難であった。そこで、部分的に合成した数個の点群を全体形状に位置合わせする段階では、幾何変換の行列係数を手作業で入力した。このように生成した全体形状を表す点群データに GPT（Greedy projection Triangulation）でメッシュを生成し、最終的な３次元モデルを完成した。

[1] Kinect は、米国 Microsoft Corporation の米国及びその他の国における登録商標または商標である。

[2] PCL は 3 次元点群データを扱うオープンソースソフトウエアで BSD ライセンスにて配布されている。

目次

第１章　序論

３次元モデルは、さまざまな工業製品や人体などの立体物の形状を、コンピュータで処理可能なデジタルデータとして表現したものである。機械部品の形状や工業製品の外観デザインあるいはゲームキャラクタなどの３次元モデルをコンピュータで設計する技術をCADや3DCGとよぶ。CADや３DCGのソフトは広く実用化されているが、その操作は複雑で、自由に３次元形状を設計できるようになるには熟練が必要である。そのため、立体物を設計したり製造することは、高度な技術と製造装置をもつ企業だけが可能なことである。

立体物の製造装置に関して、最近、安価な３D プリンタが実用化された。これに伴って、３次元モデルの利用が急速に広まると期待されている。しかし、３次元モデルの設計作業には、CAD ソフトに熟練することや立体物の造形に関する知識が必要である。３次元モデルをゼロから設計したり生成することは、かなり困難な作業であると言える。この問題を軽減するために、すでに存在する立体物の３次元形状をコピーし、それを元に、最終的な３次元モデルを設計・生成するという手法が考えられる。この技術はメカニカル・リバースエンジニアリングとよばれる。この技術は、立体物の形状を測定する技術と、測定データを３次元モデルに変換する技術、の２つ技術から構成される。

立体物の形状を測定する技術は、一般に３次元計測技術とよばれる。その技術を組込んだ測定装置を３次元入力装置あるいは３D スキャナとよぶ。３D スキャナには多くの方式・製品が存在するが、その多くは高価である。一方、マイクロソフトがゲーム機 Xbox 用のヒューマンインタフェースデバイスとして発表した Kinect も、測定装置としては３D スキャナである。Kinect は 3D スキャナとして非常に安価であるので、これをメカニカル・リバースエンジニアリングのための入力装置として利用することができれば画期的なことである。

立体物を測定する３D スキャナが出力する生データは、立体物の表面形状を離散的な点の集合によって表現したものである。一つ一つの点は３次元座標に対応しており、そのような点の集合を３次元点群あるいはポイントクラウドとよぶ。３次元点群は立体物の形状を正確に表現したデジタルデータではあるが、CAD ソフトや3DCG ソフトによる形状データの加工や編集には適さない。そこで、３次元点群データをこれらのソフトウエアで扱うことが可能な３次元モデルに変換する必要がある。３次元モデルの表現形式として、ポリゴンパッチとパラメータ曲面が代表的なものである。本論文では、三角パッチで表現された３次元モデルに変換することを目標とする。

Kinect は３次元点群データを出力する装置であるが、同様な３次元点群データを出力する装置として、レーザレンジファインダやレーザーレーダーあるいは LIDAR（Light

Detection and Ranging）とよばれるものがある。これらの装置の用途は自動車の前方監視や移動ロボットの環境監視である。これらに利用することを主目的としてオープンソースソフトウエア PCL（Point Cloud Library）が、2010 年頃から開発されていた。同じ時期に Kinect がリリースされたことによって、PCL には Kinect 用のインタフェースや３次元モデリングに利用可能な機能が開発されてきた。PCL は、３次元モデリングに特化したライブラリではないが、この目的のためにも非常に利用価値が高いものである。

このような背景から、Kinect と PCL を用いることで、安価な装置でありながら高機能な立体物の３次元モデリングシステムを開発することが可能であると考えた。本研究の目的は、このアイデアを実証することである。

第 2 章　PCL の機能

2.1　PCL の概要

PCL（Point Cloud Library）は 3 次元点群データを扱うオープンソースソフトウエアである[1]。PCL にはフィルタリング、特徴推定、サーフェス再構成、レジストレーション、モデルフィッティング、セグメンテーション等のアルゴリズムが含まれている。PCL は BSD ライセンスでリリースされており、誰でも使用できるライブラリである。PCL は関係する機能の集まりごとにモジュール化されている。図 2.1 に全てのモジュールを示す。これらのモジュールに含まれるライブラリを使うことによって、Kinect で取得した 3 次元点群データに高度な処理を加えることが可能である。

今回使用する主なアルゴリズムは、外れ値除去、初期位置合わせ、精密な位置合わせ、スムージング、ポリゴンメッシュ処理である。ここでは、外れ値除去やダウンサンプリングを行うフィルタモジュール、位置合わせを行うレジストレーションモジュール、ポリゴンメッシュによる 3 次元モデルを生成するサーフェイスモジュールについて説明する。

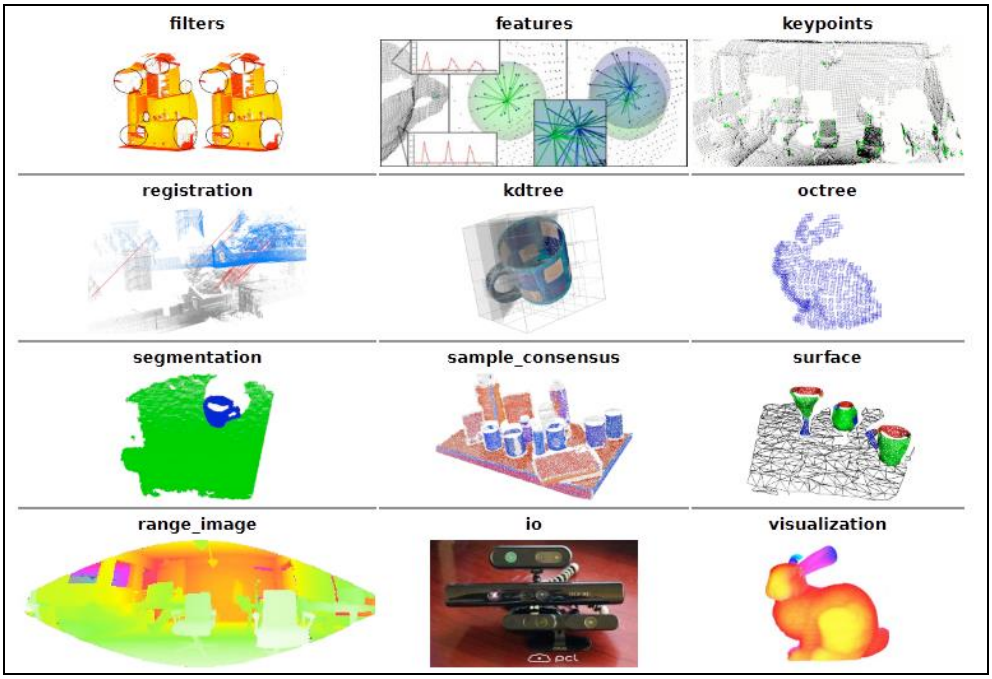


図 2.1 PCL のモジュール

2.2　３次元モデリングのための機能

2.2.1　フィルタモジュール

外れ値除去について説明する。Kinect で取得した３次元点群データには外れ値とノイズが含まれる。外れ値は真の測定値から大きく離れた誤計測値のことであり、立体物の端面がその背景部分と中間的な値として計測される場合などに発生する。ノイズは真の測定値に加わる微小な変位のことである。ノイズは統計的な性質を持っており、正規分布で近似できることが多い。測定データに外れ値が含まれていると、位置合わせなどの後処理が難しくなる。そこで、前処理の一つとして外れ値を除去する。これによって、対象の立体物表面から離れている３次元点群データを除去することができる。

ボクセルグリッドフィルタについて説明する。Kinect は一回の測定によって、640×480 画素の点群を取得する。立体物の全体を計測するためには、この 10 倍程度の点群を扱う必要がある。パソコンで、これだけのデータ量を処理すると大幅に時間が掛かってしまう。点群の密度を下げるためのダウンサンプリング処理を行ために、ボクセルグリッドフィルタを適用する。点群の密度を下げることで、後述の位置合わせ処理などを高速化することができる。

2.2.2　レジストレーションモジュール

Kinect を用いて、一つの立体物の全体形状を測定するためには、その物体を複数の視点から測定する必要がある。複数の視点から取得した３次元点群データは、そのままでは、お互いの位置関係が正しくないため、全体として一つの物体の形状を表現しているとは言えない。そこで、まず SAC-IA（Sample Consensus Initial Alignment）関数を用いて、複数の３次元点群データを粗く初期位置合わせさせる。

初期位置合わせの後、精密な位置合わせを行う。PCL では、ICP（Interative Closest Point）アルゴリズムを使って、3 次元点群データ間の精密な位置合わせを行う。このアルゴリズムは、２つの 3 次元点群データ間の重なり部分を検出し，繰り返し処理によって、正確に対応付ける。この時、初期的な位置合わせに間違いがあると、正しく収束せず、位置合わせに大きなずれが生じる。このズレを無くすためには、初期位置合わせもある程度、精度良く行う必要がある。

2.2.3　サーフェスモジュール

MLS（Moving Leas Squares）アルゴリズムは、移動最小二乗近似による３次元点群データの平滑化処理である。これは、各点の近傍を定義する範囲を指定し、その範囲内の点群データに最小二乗法を適用することでノイズを除去する。本研究の実行手順上、位置合

わせ前にノイズを除去し、点群をなめらかにすることが望ましい。MLS は近傍を定義する変数 setSearchRadius に値を設定することで、ノイズ除去効果に強弱をつけることができる。値が大きければ広い範囲になり、小されば狭い範囲になる。図 2.2 は MLS を適用する前と適用した後の例である。

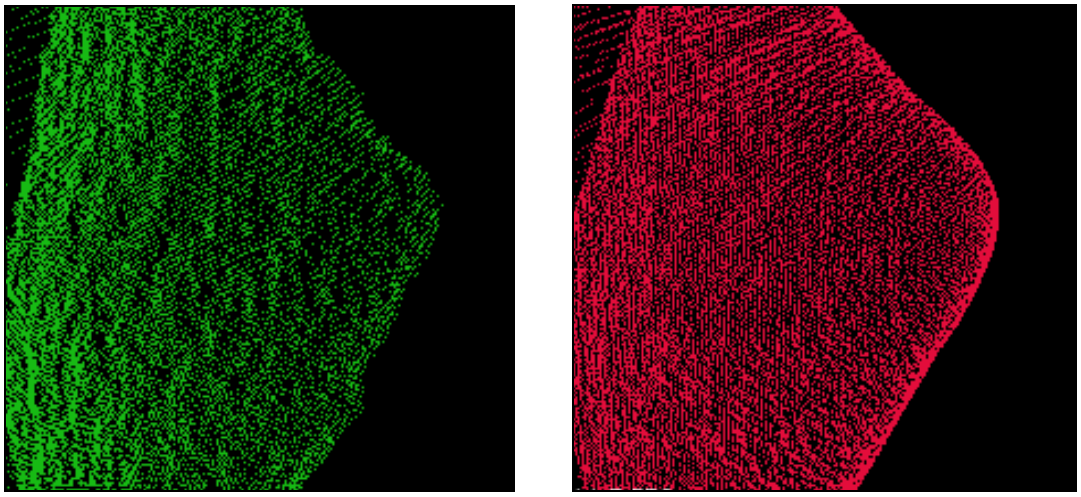


図 2.2 MLS によるスムージング前とスムージング後

GTP（Greedy projection Triangulation）アルゴリズムを用いてメッシュを生成する。このアルゴリズムは、メッシュを生成する点をリストに保持し、メッシュの生成が可能な点と点との接続を行う。この手順を繰り返すことによって、点群全体にメッシュを生成する。GTP は、１つの点群データ、または、位置合わせを行った点群データを処理し、構造化されていない点に対処することが可能である。表面が局所的に滑らかである場合や、あらかじめスムージング処理を適用することによって、穴の少ないメッシュが生成できる。

第 3 章　Kinect による立体物の測定

3.1　Kinect の測定原理

Kinect の 3D センサは、赤外線レーザーを光源とするプロジェクタで単一のパターン光を照射し、対象物体に投影されたパターンを赤外線カメラで撮影する。図 3.1 に示すように、パターン光の各部の照射角度とその部分をとらえたカメラ画像の見込み角度、およびプロジェクタとカメラの距離を用いて、その部分の 3 次元的な位置を三角測量で計算する。

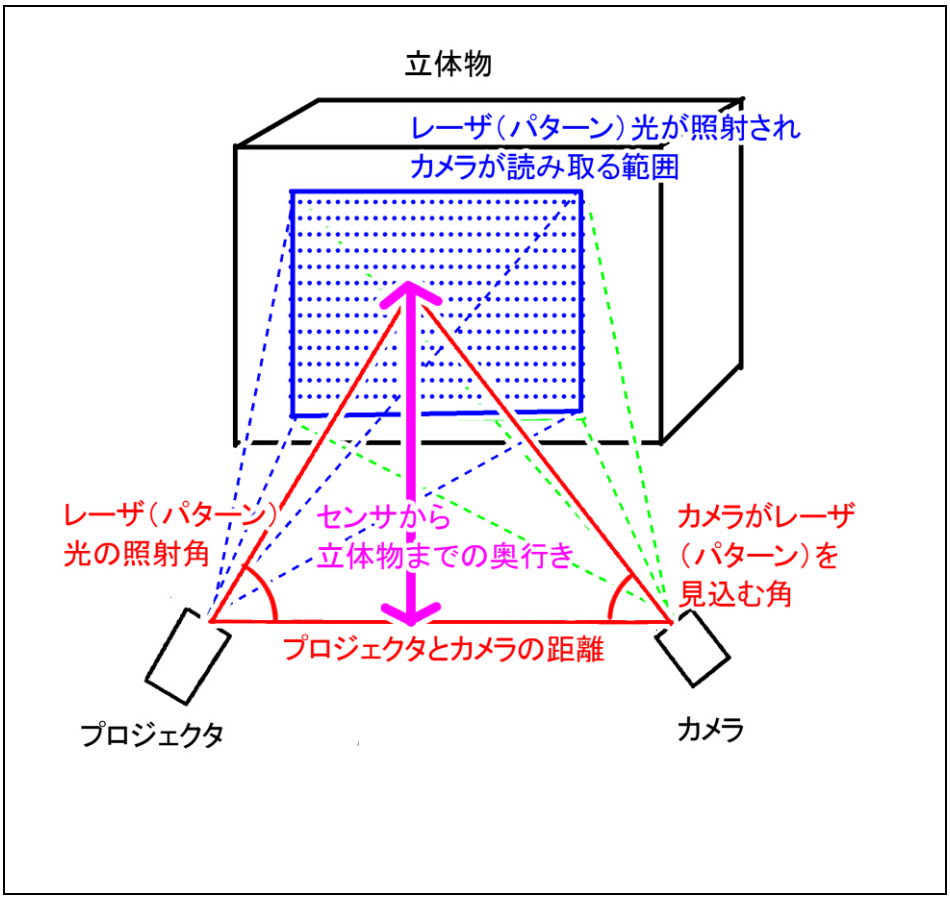


図 3.1　Kinect による 3 次元計測の原理

3.2　Kinect の測定精度

Kinect には測定誤差が生じる。その測定精度を検証した。被写体として白と黒の画用紙を用い、画用紙を平面にするため、壁に貼付けた状態で測定した。また環境光の影響を確認するため、室内照明有りと無しで測定した。測定状態を図 3.2、図 3.3、図 3.4、図 3.5 に示す。

図 3.2　撮影距離 1m・白画用紙・照明有り

図 3.3　撮影距離 2m・白画用紙・照明有り

図 3.4　撮影距離 3m・白画用紙・照明有り

図 3.3　撮影距離 4m・白画用紙・照明有り

Kinect で撮影する際、画用紙がおおむね撮影領域中央に入るように設置する。距離ごとに取得するデータを比較しやすくするために、抽出範囲を設定する。その抽出範囲は、4m 離れた距離で撮影した画用紙の大きさに合わせる。図 3.6、図 3.7 のように赤く表示した部分が抽出範囲である。入力画像上で抽出範囲の大きさを揃えることで、被写体までの距離によらず、抽出される点の個数がほぼ同じになる。抽出した点群データの例を図 3.8 示す。

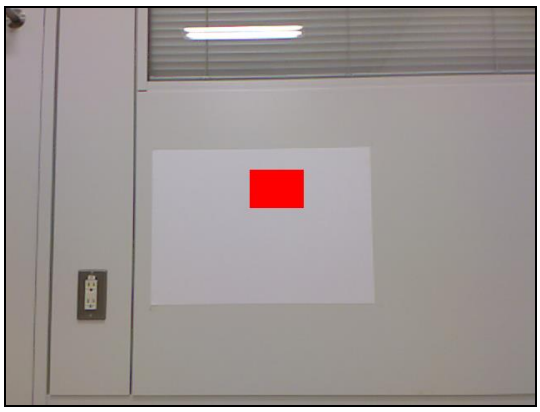


図 3.6　撮影距離 1m での抽出範囲

図 3.7 撮影距離 4m での抽出範囲

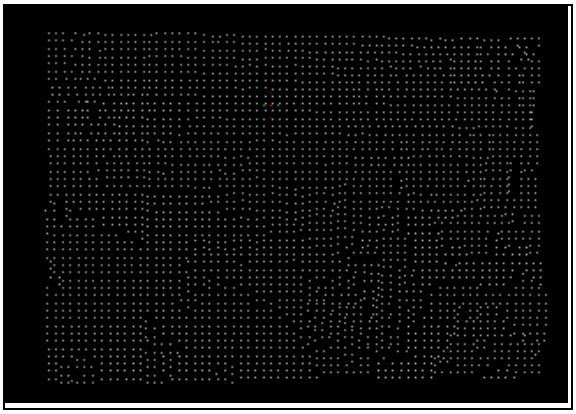


図 3.8　抽出した点群データの例

この点群データは、壁面に貼付けた画用紙の表面を表現する。したがって、平面状になっていると予測される。しかし、この点群を 90° 回転させて画用紙面を横から観察すると、図 3.9 のように、5～6 の層が生じていた。この層は Kinect の奥行き方向の測定誤差に起因するものであると考えた。

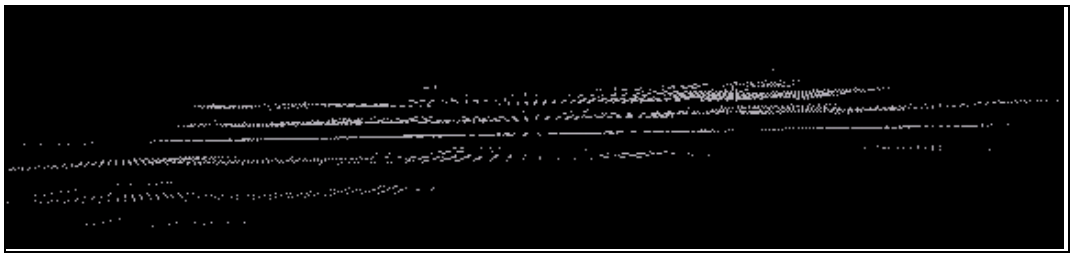


図 3.9　画用紙の点群データを水平に表示

奥行き方向の測定に生じる誤差として、ランダム誤差と量子化誤差が考えられる。まずランダム誤差について検討する。ランダム誤差は、測定値が真の値から乱数的な性質を持つ微小な値だけ変化する誤差のことである。測定系にはランダム誤差が不可避であるから、Kinect の奥行き測定値にもなんらかのランダム誤差が生じている。しかしランダム誤差は図 3.9 の層状の測定結果を生じるものではない。

量子化誤差は、測定装置の解像度が有限の桁数を持つことによるものである。Kinect は奥行き方向の測定値を 13bit で表現するため、その量子化誤差により点群の位置が丸められてしまう。表 3.1 に 1000mm の距離にある白画用紙を測定した点群データの一部を示す。奥行きを表す z 値に注目すると、1.0780001 と 1.0810001 の２つの数値が現れる。したがって、奥行き方向の量子化誤差が約 3mm に相当しており、図 3.9 の層間の距離も約 3mm と考えられる。

ここで、点群の水平・垂直方向の間隔についても検討する。今回使用している Kinect の解像度は 640×480 であり、水平視野角は 57°、垂直視野角は 43°となっている。図 3.1 に示す三角測量による測定で、撮影距離が 1000mm の場合の横 1 画素分の幅は 1.69mm である。縦 1 画素分の幅は 1.64mm である。撮影距離と水平視野角、解像度を使って、この数値を計算した式を以下に示す。

tan(57/2)=X/1000	tan(43/2)=X/1000
X=1000×0.549	X=1000×0.3939
X=542.9mm	X=393.9mm
542.9×2=1085.8	393.9×2=787.8
1085.8/640=1.69mm	787.8/480=1.64mm

したがって、1000mm の距離にある被写体を測定した３次元点群は、水平・垂直方向におよそ 1.6～1.7mm の隙間ができるはずである。実際に点群データの数値を確認したところ、撮影距離が 1000mm の撮影物に対し、水平・垂直方向に 2mm 程度の隙間ができていることを確認した。

表 3.1　白画用紙・照明有り・撮影距離 1m に対する点群の座標値の例

X	Y	Z
-0.06570667	-0.073920004	1.0780001
-0.063653335	-0.073920004	1.0780001
-0.061600003	-0.073920004	1.0780001
-0.059546668	-0.073920004	1.0780001
-0.057653338	-0.074125722	1.0810001
-0.055594292	-0.074125722	1.0810001
-0.05338667	-0.073920004	1.0780001
-0.051333334	-0.073920004	1.0780001
-0.049280003	-0.073920004	1.0780001
-0.047226667	-0.073920004	1.0780001
-0.045173336	-0.073920004	1.0780001
-0.043120001	-0.073920004	1.0780001
-0.041066669	-0.073920004	1.0780001
-0.039121907	-0.074125722	1.0810001
-0.037062861	-0.074125722	1.0810001
-0.034906667	-0.073920004	1.0780001
-0.032944765	-0.074125722	1.0810001
-0.030800002	-0.073920004	1.0780001
-0.028746668	-0.073920004	1.0780001
-0.026693335	-0.073920004	1.0780001
-0.024640001	-0.073920004	1.0780001
-0.022586668	-0.073920004	1.0780001
-0.020533334	-0.073920004	1.0780001
-0.018480001	-0.073920004	1.0780001
-0.016426668	-0.073920004	1.0780001

画用紙の白黒、照明の有無、測定距離の違いによる誤差の性質を解析した。撮影距離は500mm、650mm、750mm、1000mm、2000mm、3000mm、4000mmに設定した。壁に貼り付けた画用紙部分の点群に平面をフィティングし、推定された平面と測定された点の垂直距離を誤差として評価した。図3.10に、横軸を撮影距離、縦軸を誤差の標準偏差としたグラフを示す。標準偏差が大きいほど測定精度が低く、標準偏差が小さいほど測定精度が高い。全体として見ると、撮影距離が短いほど測定値の精度が高い。

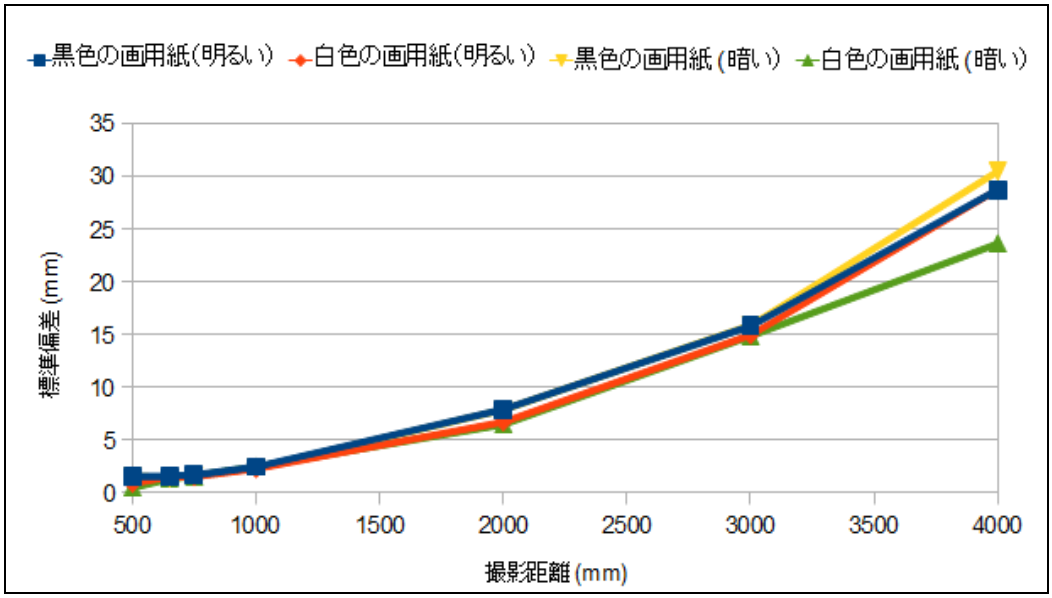


図3.10　画用紙の白黒、照明の有無、距離違いによる誤差の比較

まず、500mmと4000mmに関して、点群データの個数が他のデータより少なくなっていた。そもそもKinectは750mm～3500mmでの撮影を推奨している。そのため、750mm以下、および3500mm以上の距離で撮影すると、Kinectは正確に測定することが難しいと考えられる。図3.10の1000mmから3000mmの範囲における4つの標準偏差を比較すると、それほど差がない。この範囲であれば、被写体の色と室内の変化が測定値に大きな影響を与えないことがわかる。4000mmに注目すると、被写体が白い画用紙で室内照明を消灯している場合に、他の3つの場合より標準偏差が小さい。このことから、被写体までの距離が4000mmの場合には、暗い状態で白い被写体を測定すると精度が良くなると言える。

Kinect で取得した点群データに、PCL の MSL アルゴリズムを適用してノイズを除去し、スムージング処理を施した。図 3.9 の点群データにスムージング処理を適用した結果を図 3.11、図 3.12、図 3.13 に示す。これらの図から、スムージングを調整するパラメーターの値を大きくすることによって、平面を測定した点群が滑らかになることがわかる。

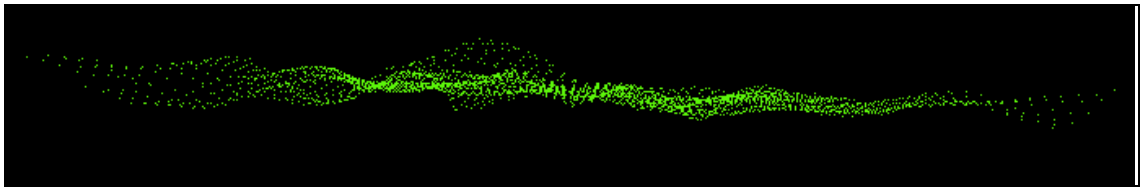


図 3.11　距離 1000mm・白画用紙・照明有り・MSL アルゴリズム（パラメータ値 0.01）

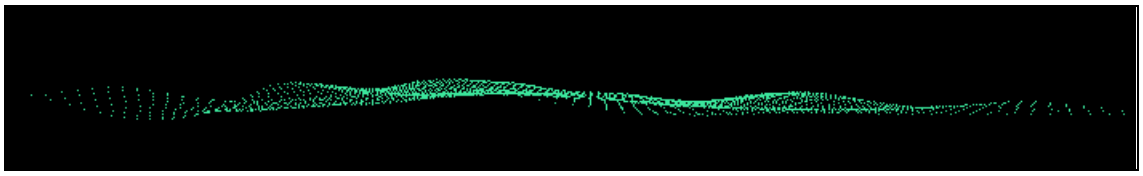


図 3.12　距離 1000mm・白画用紙・照明有り・MSL アルゴリズム（パラメータ値 0.03）

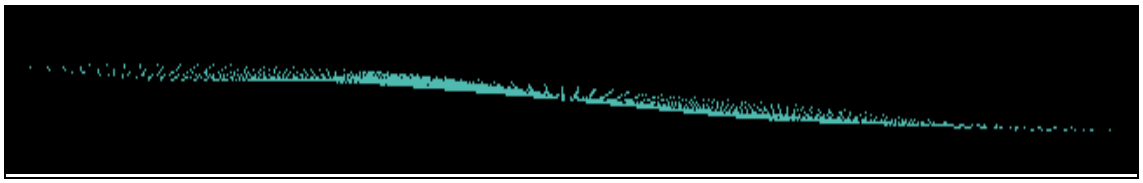


図 3.13　距離 1000mm・白画用紙・照明有り・MSL アルゴリズム（パラメータ値 0.06）

1000mm に置いた白い画用紙を照明有りで測定した場合について、横軸をスムージングのパラメータ、縦軸を誤差の標準偏差としたグラフを図 3.14 に示す。パラメータの値が 0.2 程度までは、誤差の標準偏差が 0.5 に漸減し、それを超えると標準偏差がほとんど減少しない。この理由は、測定データに装置固有の歪みが生じているためである。図 3.13 のように強くスムージング処理を施すと全体として滑らかになるが、どれほどスムージングを強めても決して平面にはならない。過度にスムージングを施して湾曲した面と、点群にフィティングした平面の誤差が除去できない誤差として残る。

各距離の点群データにスムージングを施した効果を図 3.15 に示す。スムージングのパラメータ設定値は 0.2 とした。このように、Kinect で取得したデータには誤差が含まれているが、スムージングを適用することで誤差を減少させることができる。

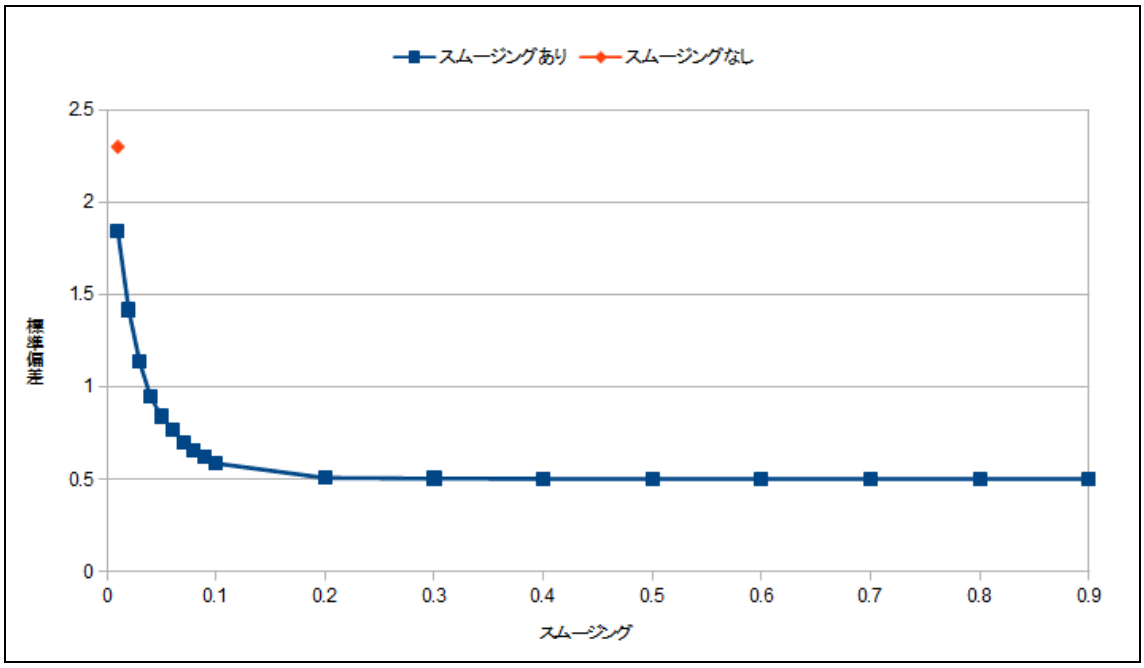


図 3.14　誤差の標準偏差とスムージングパラメータの関係

（距離 1000mm・白画用紙・照明有り・MSL アルゴリズム）

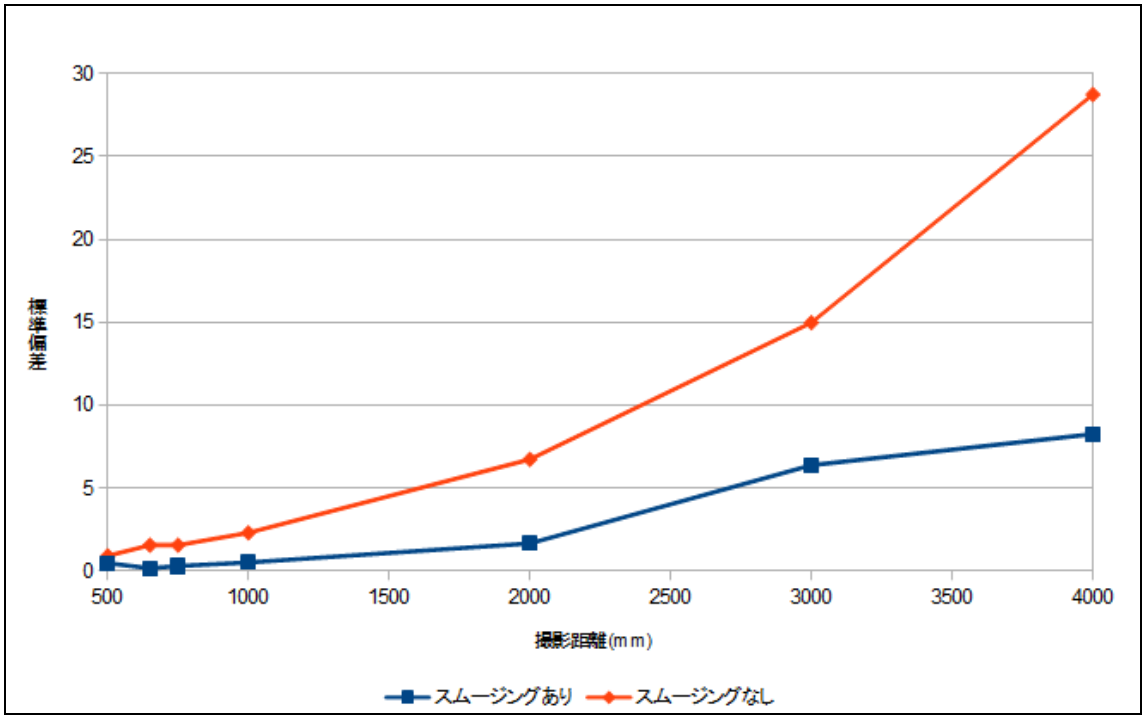


図 3.15　スムージングを適用することによる誤差の減少

（距離 1000mm・白画用紙・照明有り・MSL アルゴリズムのパラメータ 0.2）

第 4 章 Kinect と PCL を用いたモデリング

第２章で、３D モデルを作成するために利用する PCL のモジュールについて説明した。ここでは、実物体を使って、具体的な３D モデリングの手順を検証する。

4.1　３次元モデリングの処理手順

Kinect で被写体を全周から撮影する。一回の撮影では被写体の一部しかカバーできないため、被写体の全周を 12 枚に分割して撮影する。このとき、隣り合う位置での撮影に適度な重複領域を持たせる。これらの撮影データによって、12 個の３次元点群データを獲得する。取得した３次元点群データから背景を取り除き、メインとなるモデルだけを抽出する。ここから PCL の処理を適用する。第一に３次元点群データの外れ値を削除する。第二にスムージング処理を行う。第三にボクセルグリッドフィルタ処理で点群の密度を減らす。第四に初期位置合わせを行い、大まかな位置を合わせる。第五に精密位置合わせを行う。第六に位置合わせ後の点群データにスムージング処理とボクセルグリッドフィルタを適用する。第四から第六の処理を隣接する点群に繰り返すことで、全体形状に近づける。最後に、完成した位置合わせ後のデータにポリゴンメッシュ処理を行う。

4.2　立体物の測定

被写体の例として、プラスチック製のサンタクロース像を用いた。サンタクロース像を図 4.1 に示す。サンタクロース像は縦 76cm、横 35cm の大きさである。サンタクロース像を Kinect で撮影し、３次元点群データを獲得する。Kinect とサンタクロース像の撮影距離を約 1m に設定した。Kinect は図 4.1 のカラー画像とともに、図 4.2 の３次元点群データを取得する。図 4.2 は取得した点群座標にカラー情報を付与した３次元点群データを若干異なった視点から表示したものである。

図 4.1　サンタクロース像のカラー画像

図 4.2　サンタクロース像の点群データ

4.3　測定データに対する前処理

Kinect で取得する点群データには被写体と背景が含まれる。被写体のモデリングを行うために、背景を切り取る必要がある。被写体のみのデータを抽出するために、カラー画像を使って背景を取り除く。作成したプログラムは、カラー画像上でマウスを使って対象物と背景の境界を指定する。対象物を囲む領域をマウスで指定し、そこを図 4.3 のように赤色で塗りつぶす。そして、赤く描画した部分を抽出することで、背景を取り除く。

Kinect による 1 回の測定では対象物の一部の形状しか取得することができないので、全周から撮影した点群データが必要である。今回は、被写体の全周を 30° 間隔で回転させながら 12 個の点群を取得した。図 4.4 に、それらから背景を取り除いたものを示す。この 12 個の点群データを用いて 3 D モデリングを行う。

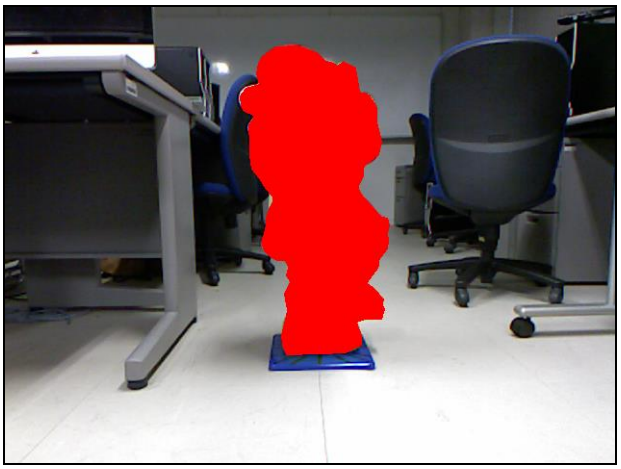
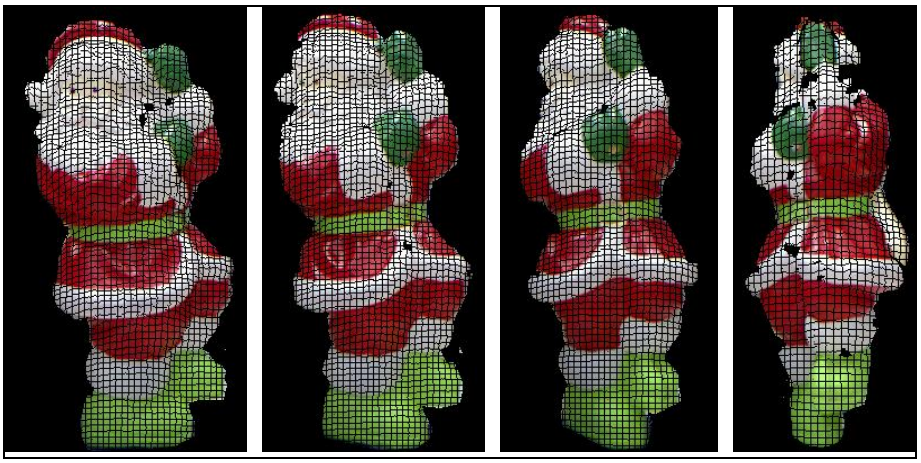
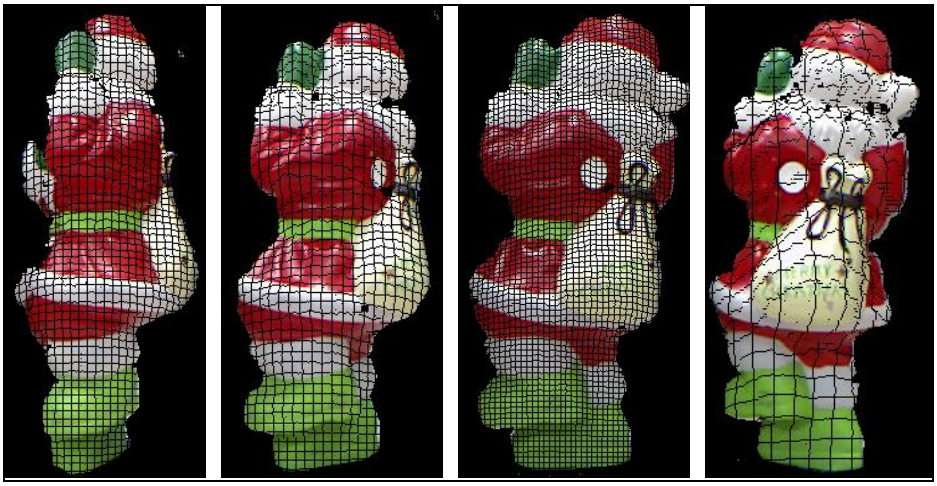


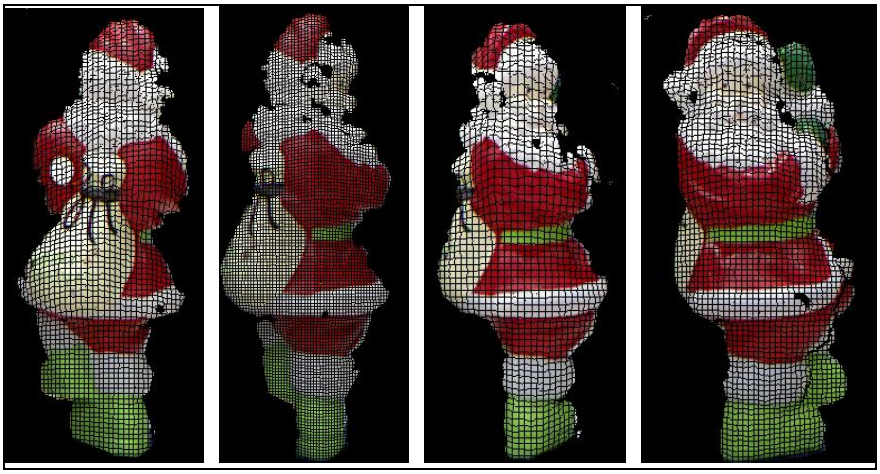
図 4.3 抽出範囲を設定



(a)0　(b)30°　(c)60°　(d)90



(e)120°　(f)150°　(g)180°　(h)210°



(i)240°　(j)270°　(k)300°　(l) 330°

図 4.4　背景を取り除いたサンタクロースのデータ

4.4　外れ値除去とダウンサンプリング

背景を取り除いた点群データには外れ値が含まれる。外れ値を取り除くために外れ値除去フィルタを適用する。図 4.4(a)の点群データの外れ値を除去した例を図 4.5 に示す。なお、外れ値の多くは対象物と背景の境界に生じる。距離の段差が存在する境界では、その位置を正確に測定することが難しく、外れ値や大きな誤差となる。

次いで、スムージング処理を施してノイズを低減した後、処理時間を削減するためにダウンサンプリングを行う。ダウンサンプリングを行うために、ボクセルグリッドフィルタを適用する。ダウンサンプリングした例を図 4.6 左に示す。見えやすくするために、点を膨張させた例を図 4.6 右に示す。

図 4.5　外れ値除去前と外れ値除去後

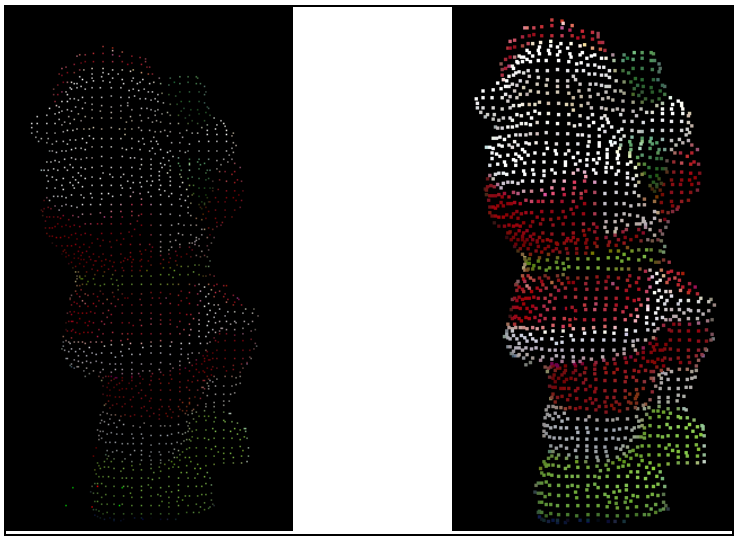


図 4.6　ダウンサンプリング

4.5　複数データの位置合わせ

外れ値除去、スムージング、ダウンサンプリングを(a)~(l)の点群データに適用した後、位置合わせ（レジストレーション）を行う。レジストレーションを行うためには、適度な重複領域を持った二つの点群データが必要である。レジストレーションを行い、点群データを重ねていくことで欠けている形状を追加させる。レジストレーションは初期位置合わせと精密位置合わせを用いて処理を行う。初期位置合わせにより、大まかに位置合わせをした後、精密位置合わせにより精度良く点群の座標位置を合わせる。１回位置合わせをする度に、重なり部分に生じるダブルウォールを除去するためにスムージングを行う。(a)と(b)、次いで(a)と(b)と(c)、(a)と(b)と(c)と(l)のように、隣り合っているデータを順番に位置合わせし、点群がカバーする対象物の領域を拡張する。図 4.4(a)の点群データと(a)・(b)・(c)を位置合わせした点群データを左に回転させ、表示した例を図 4.7 に示す。図 4.7 の右画像に注目すると、サンタクロースの袋の部分が追加されていることが観察できる。図 4.4(a)の点群データと(a)・(b)・(c)を位置合わせした点群データを右に回転させ、表示した例を図 4.8 に示す。図 4.8 の右画像に注目すると、欠けている手元の部分や足の部分が追加されていることが観察できる。

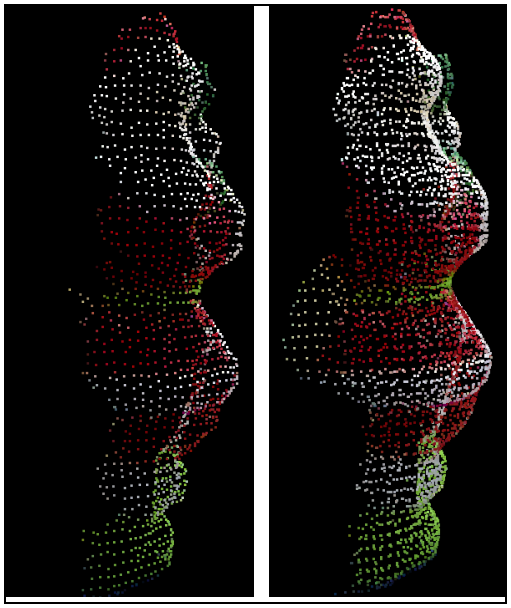


図 4.7　左に回転した点群データ
左：位置合わせなし
右：位置合わせあり

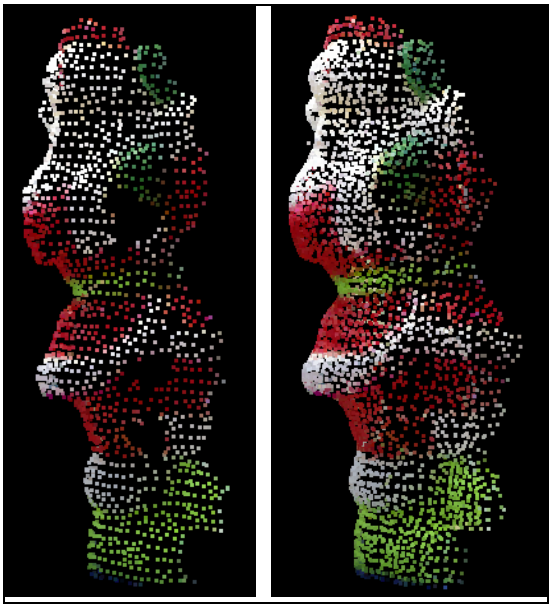


図 4.8　右に回転した点群データ
左：位置合わせなし
右：位置合わせあり

隣接する点群を増やしていったが、ここで問題が発生した。位置合わせを重ねるにしたがって、図 4.9 のように位置合わせに失敗するケースが多発した。図 4.9 左は緑部分の手が４つになっており、ずれた状態になった。右は黄緑の靴が上と下になっており、上下が逆さまになった。４枚程度までの位置合わせはおおむね成功するが、それを超えると、多くの場合に位置合わせに失敗する。初期位置合わせの組合せが複雑化することで、初期位置合わせに失敗することが主な原因と推察される。

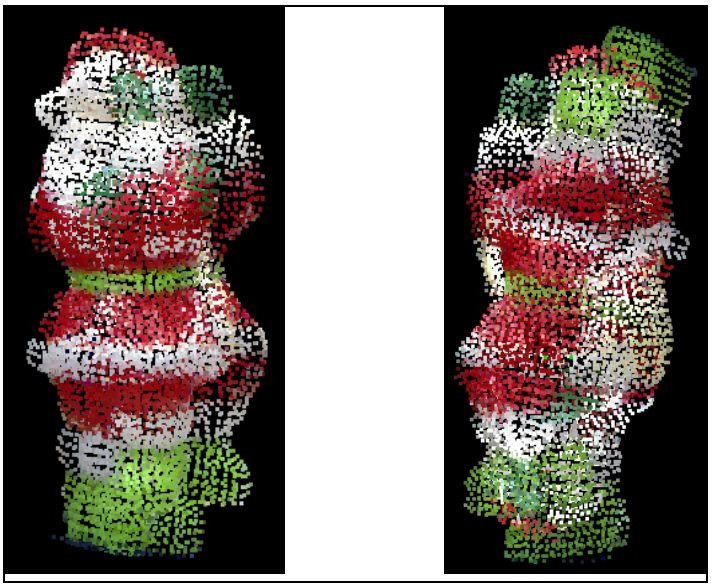


図 4.9　失敗するケース

左：(ずれている状態)、右：(上下逆さまの状態)

位置合わせに失敗する対策として、点群間の座標変換に用いる行列の係数を手作業で設定することにした。手作業で設定した変換で点群を合成し、その合成精度を目視で確認した。この作業を繰り返すことで、可能な限り精度良く点群を合成した。(a)・(b)・(l)を位置合わせした点群と(f)・(g)・(h)を位置合わせした点群を、手作業で設定した変換行列で位置合わせした例を図 4.10 に示す。(a)・(b)・(l)を位置合わせした点群によって、正面側がおおむねカバーされ、(f)・(g)・(h)を位置合わせした点群によって、背中側がおおむねカバーされる。この２つの点群を位置合わせ合成することで、全体を表現する点群を生成することができた。

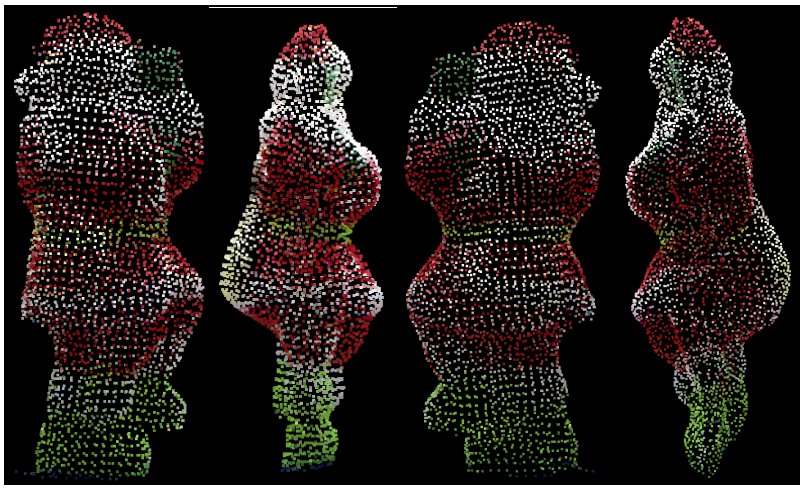


図 4.10　手作業で設定した変換行列による位置合わせの例

左：(正面)、左真ん中：(右向き)、右真ん中：(真後ろ)、右：(左向き)

4.6　スムージング

Kinect で取得した点群データにはノイズが生じている。点群データのノイズを軽減させるためにスムージング処理を行う。また、２つの点群を位置合わせした点群データにはダブルウォールが生じる。ダブルウォールは、位置合わせによる極微小なズレによって、点群が２重に整列した状態になっている。ダブルウォールが存在していることによって再度行う位置合わせやサーフェス生成が難しくなる。スムージング処理でこれを解決することができる。

4.7　ポリゴンメッシュよるサーフェス生成

ポリゴンメッシュ処理によりサーフェス生成を行う。これにより点群データをメッシュデータに変換する。 図 4.4(a)の点群データに直接ポリゴンメッシュ処理を適用したメッシュデータを図 4.11 の左に示す。外れ値フィルタ、ボクセルグリッドフィルタ、スムージングを適用した点群から生成したメッシュデータを図 4.11 の右に示す。図 4.11 左は、細かいしわが発生した様になっており、右は滑らかになっていることが確認できる。図 4.11 のメッシュデータを 90°回転させ、サンタクロース像の左腕部分を拡大したものを図 4.12 に示す。図 4.12 左は、ポリゴンの向きが不規則になっている。これがしわのように見えた原因である。図 4.12 右はポリゴンの向きが滑らかに変化している。

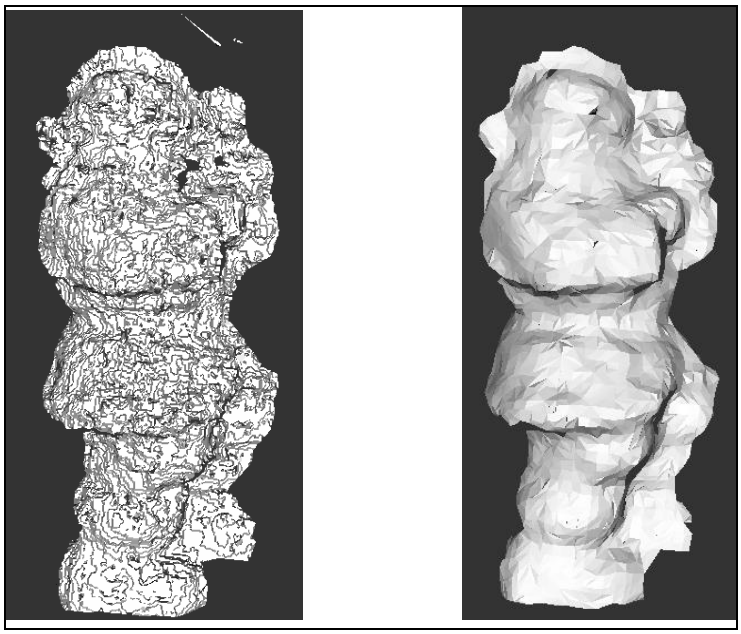


図 4.11　図 4.4(a)から生成したメッシュデータ

左：(外れ値フィルタ、ボクセルグリッドフィルタ、スムージング）なし

右：(外れ値フィルタ、ボクセルグリッドフィルタ、スムージング）あり

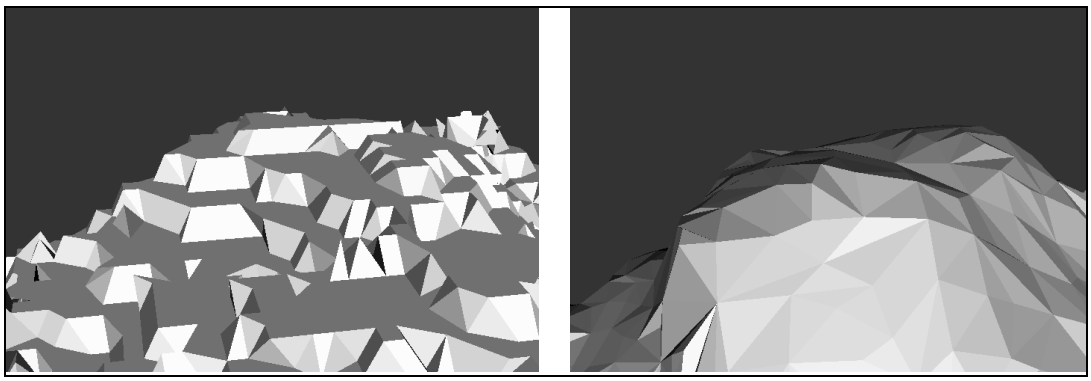


図 4.12　メッシュデータ（サンタクロースの左腕部分を拡大）

図 4.11 右のメッシュデータを右に回転させ、表示した例を図 4.13 左に示す。図 4.13 左の手元と足元に注目すると、穴の開いた状態になっており、メッシュが生成されていないことが観察できる。この穴に関しては、MLS アルゴリズムの設定を変更することで、埋めることができる。図 4.4(a)の点群データにフィルタリングを行い、ポリゴンメッシュ処理の設定を変更し、適用し直したメッシュデータを図 4.13 右に示す。ポリゴンメッシュ処理のパラメータ値を表 4.1 に示す。図 4.13 左と右を比較し、右の手元と足元を注目すると、メッシュが生成されていることが確認できる。しかし、手と頭に注目すると、手と頭の間部分にもメッシュが生成されている。したがって、ポリゴンメッシュ処理で無理やり穴を埋める設定にすると、輪郭部分に余計なメッシュが追加されてしまう等の影響が生じる。綺麗なメッシュデータを生成するには、点群の部分ごとにフィルタリング処理のパラメータを変更するなどの、適応的な処理が必要である。

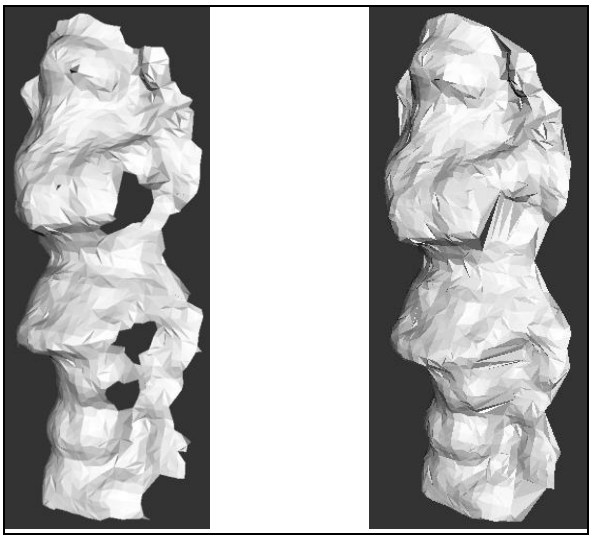


図 4.13　図 4.4(a)に対するメッシュデータ（MLS の設定による形状の比較）
左：メッシュができていない部分が多い、右：余計なメッシュが追加されている

表 4.1 ポリゴンメッシュ処理の設定値

メソッド	図 4.13 左	図 4.13 右
setSearchRadius	0.3	0.3
setMu	2.5	2.75
setMaximumNearestNeighbors	500	1000
setMaximumSurfaceAngle	M_PI/4	M_PI/1
setMinimumAngle	M_PI/18	M_PI/18
setMaximumAngle	2*M_PI/3	2*M_PI/2

フィルタリングを適用した(a)、(b)、(c)の３つの点群データに対してレジストレーションを行い、ポリゴンメッシュを適用したメッシュデータを図 4.14 に示す。図 4.13 と図 4.14 を比較すると、図 4.14 は穴が塞いでいる上に、余計なメッシュが追加されていない。したがって、一つの点群からメッシュを生成するより、複数の点群データによる位置合わせ後の点群データでメッシュを生成する方が、より正確にモデルの形状を表現できる。

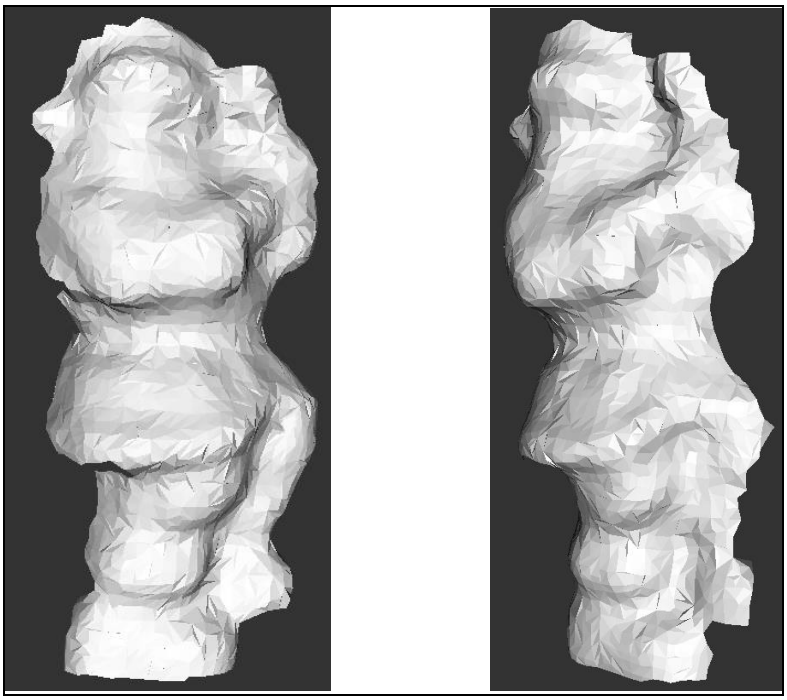


図 4.14　図 4.4 の(a)(b)(c)にフィルタリングと位置合わせを行った点群から
生成したメッシュデータ
左：正面を表示、右：左回転し表示

モデル全体の形状を表現したメッシュを生成するには、モデル全体の形状を表現した点群が必要になる。図 4.10 の点群を、モデル全体を表す点群としてメッシュデータを生成した。図 4.15 に、生成されたメッシュを示す。図 4.4 と図 4.15 を比較すると、図 4.15 は第一に色値がなくなっている。第二に厚みが狭くなっている。第三にサンタクロース像の髭部分やベルト等の形状が確認できない。第四にメッシュデータのところどころに不自然な凸凹があり、元の形状を再現できていない。第一の理由は点群データからメッシュデータを生成する時に、色情報が削除されたかたである。第二の理由は正面部分のデータと真後ろ部分のデータを位置合わせしたため、側面データが不足したことによるものである。第三の理由はスムージング処理の効果が効きすぎている可能性がある。第四の理由はポリゴンメッシュによるメッシュ生成なので、完全な曲面を表現することができないからである。

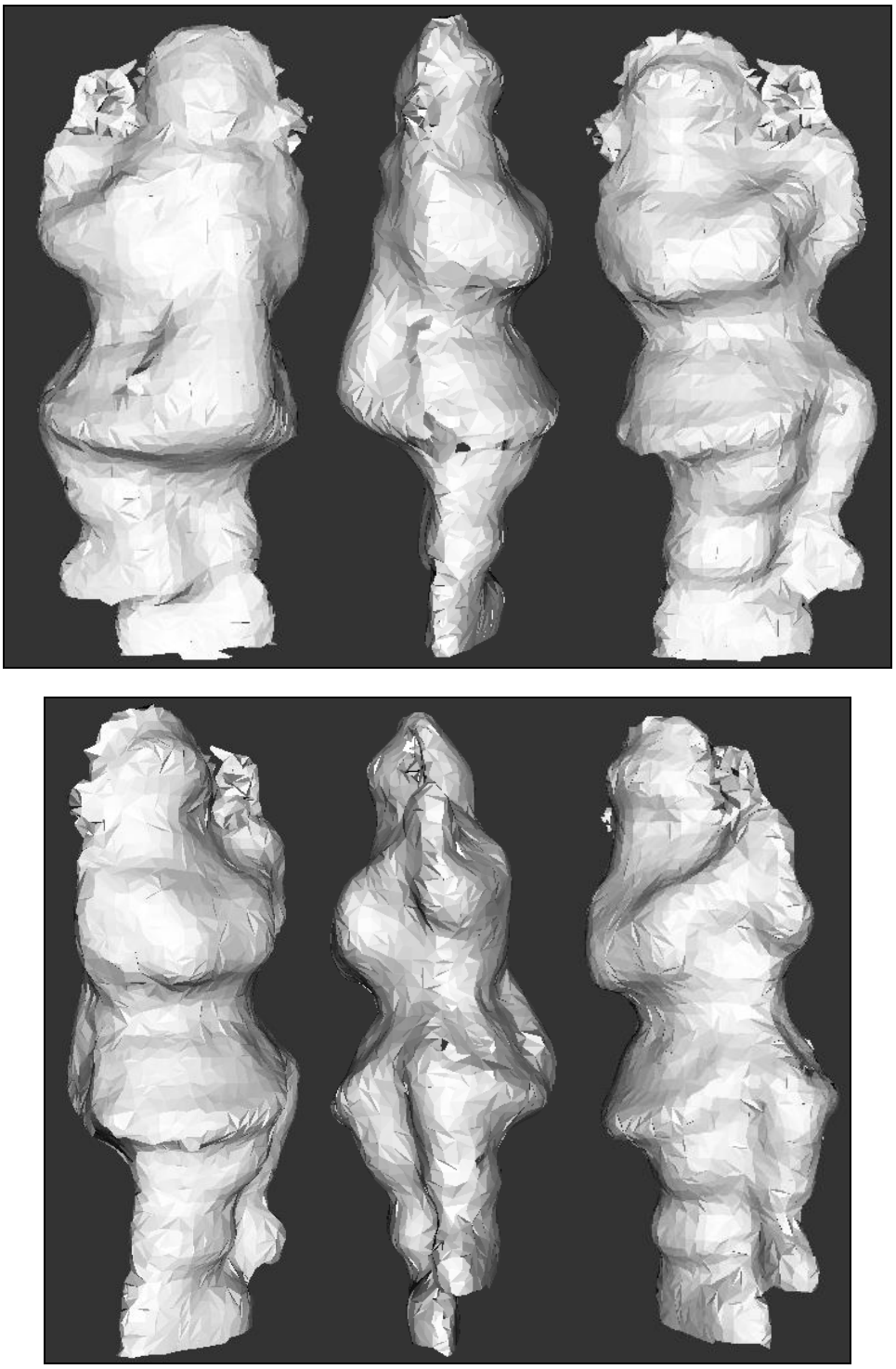


図 4.15　図 4.4(a)(b)(l)と(f)(g)(h)の点群データから生成したメッシュ
左上：真後ろ、真ん中上：右向き、右上：正面、
左下：右斜、真ん中下：左向き、右下：左斜め

4.8　考察

立体物の３D モデリングを行うために、被写体を全周囲方向から複数枚に分けて撮影した。取得した複数の点群データを、隣接する点群同士で位置合わせすることで、全体形状を表現する点群データを生成できると考えた。位置合わせを行うに当たり、隣り合う点群データとの位置合わせは成功した。しかし、位置合わせ回数が増えるにしたがって、失敗するケースが生じた。位置合わせに失敗する理由は、第一に初期位置合わせに失敗している可能性がある。第二に精密位置合わせで、精度良く位置合わせできていない。位置合わせの失敗に対処する方法として、PCL による自動的な位置合わせを使用せず、座標値を回転・移動させる変換行列の設定を手作業で行った。手作業による変換行列の設定は、処理ごとに合成精度を目視で確認しなければいけないため、合成の手間と時間がかかる。

第 5 章　結論

5.1　成果

本研究では Kinect の測定精度を調査し、次いで、Kinect と PCL を用いて実物体の 3 D モデルを生成した。

Kinect の測定精度は、被写体との距離が短いほど精度が良く、1000mm で σ=2.5mm、2000mm で σ=7.5mm、3000mm で 15.0mm 程度になった。また、750mm から 3000mm の範囲では、被写体の濃淡と照明の有無によらず、同程度の測定精度であった。測定誤差はスムージング処理を行うことで低減することが可能であるが、測定データに、装置固有の歪みが生じているため、それを除くことはできなかった。

モデリングの検証のために、モデルとなるサンタクロース像を Kinect で 30° ごとに 12 枚撮影し、それらから、PCL を用いて全体の 3 D モデルを作成することに取り組んだ。前処理として、獲得した点群データの背景を除去し、外れ値フィルタとスムージング処理とボクセルグリッドフィルタを適用した。このように準備した 12 個の点群を位置合わせ合成することで、全体形状を表現する点群を生成する計画であった。しかし、合成する点群の数が増えるにしたがい、PCL による自動的な合成が失敗した。そこで、正面とその左右 30° の 3 点群を合成した点群と。真後ろとその左右 30° の 3 点群を合成した点群を、手作業で位置合わせしたものを、全体を表現する点群とした。位置合わせした点群データにポリゴンメッシュ処理を行い、 3 D モデルを作成した。作成した 3 D モデルは blender[2]等の 3 DCG ソフトで読み込むことができ、最終的に 3 D プリンタで出力することも可能である。

5.2　課題

Kinect の測定誤差を改善する一つの方法として、静止物体を測定するのであれば、連続して撮影した生データを加算平均することが有効である。また、Kinect による 3 次元計測には幾何学的な歪みが生じている。これを改善するには、幾何学的な歪みを、あらかじめ補正することが有効である。

複数点群の位置合わせに関して、すべての点群データを PCL の機能で自動的に位置合わせすることはできなかった。対策として変換行列による位置合わせを行ったが、一回の処理ごとに合成精度を目視で確認しなければいけないため手間と時間がかかる。より簡易で正確に位置合わせができる機能を考案する必要がある。

生成されたメッシュデータには色値が含まれていない他にも、形状を正確に表現しているともいえないので、実用的に 3 D モデルにするためにはそれらを補う必要がある。

謝辞

本研究を進めるにあたり、適切な助言、機材の準備、また様々なご指導を頂きました蚊野浩教授に感謝いたします。

参考文献

[1]PCL - Point Cloud Library（PCL）， http://pointclouds.org/

[2]blender.org ， http://www.blender.org/

付録

[1]主要なプログラム

pcl.cpp	メイン関数である。PCL オブジェクトの初期化を行う。ユーザーに対してコマンドキーを受け取る。ユーザーのコマンドによって、それぞれの処理を行う。
Visualizer.cpp	点群データの読み込む、メッシュ生成データを読み込む、Kinect の動作と制御を行い点群データに変換する処理を行う。
Registration.cpp	外れ値フィルタ、ボクセルグリッドフィルタ、初期位置合わせ、精密位置合わせの処理を行う。
regidTransformation.cpp	点群データの背景除去、変換行列による位置合わせの処理を行う。
Triangle.cpp	スムージング、ポリゴンメッシュ処理を行う。
NoiseEstimate.m	ノイズ検証を行うプログラム。点群データを読み込み、誤差の平均、絶対値の平均、分散、標準偏差を計算する。

[2]PCL のパラメータ値

外れ値フィルタ　StatisticalOutlierRemoval	
SetMeanK	50
setStddecMulThresh	1.0
ボクセルグリッドフィルタ　VoxelGrid	
setLeafSize	0.015,0.015,0,015
スムージング　MovingLeastSquares	
setSearchRadius	0.03

初期位置合わせ　SampleConsensusInitialAlignment	
setCorrespondenceRandomness	15
setNumberOfSamples	5
setMaxCorrespondenceDistance	1
setMinSampleDistance	0.01
setMaximumIterations	50
精密位置合わせ　IterativeClosestPoint	
setRANSACOutlierRejectionThreshold	0.1
setMaxCorrespondenceDistance	0.1
ポリゴンメッシュ　GreedyProjectionTriangulation	
setSearchRadius	0.3
setMu	2.5
setMaximumNearestNeighbors	500
setMaximumSurfaceAngle	M_PI/2
setMinimumAngle	M_PI/18
setMaximumAngle	2*M_PI/3