

平成 28 年度卒業論文

平面近似を利用した 3 次元計測データの  
大きな欠損に対する点群修復

宮崎大学 工学部 情報システム工学科

岡 和寿

指導教員 椋木雅之

## 目次

1. はじめに .....	1
2. 点群修復 .....	2
2.1 点群修復とは .....	2
2.2 点群修復の従来研究 .....	2
2.3 点群修復手法 .....	5
3. 提案手法 .....	6
3.1 提案手法の流れ .....	6
3.2 点群データの取得 .....	7
3.3 平面上の点群データの抽出 .....	8
3.4 点群データから画像への変換 .....	9
3.5 画像修復 .....	10
3.6 画像から点群データへの変換 .....	12
4. 点群修復実験 .....	14
4.1 実験手順 .....	14
4.2 大きな欠損がある点群データ .....	15
4.2.1 使用する点群データ .....	15
4.2.2 実験結果 .....	16

4.3	3つの平面から構成される角の点群データ .....	18
4.3.1	使用する点群データ .....	18
4.3.2	実験結果 .....	19
5.	おわりに .....	21

## 1. はじめに

近年、画像修復分野の研究成果が実用レベルで利用されている。画像修復とは写真についての傷や意図せず写りこんでしまった物体など画像内の不要な部分を取り除き、取り除かれた領域（欠損領域）を自動的に補完する技術のことである。例えば Adobe 社が販売している Photoshop には、写真内の不要なもの（キズや欠損）を消去する機能がある。しかし、点群データの欠損修復に関する研究はあまりされていない。

点群データは Kinect などの 3 次元スキャナで現実世界の物体をスキャンすることで取得できる。しかし、3 次元スキャナで点群データを取得した場合、手前にある物体によって、背景に大きな欠損ができてしまう（図 1）。欠損のある状態だと現実世界と大きく異なるデータとなってしまう。そこで本研究では、手前の物体によってできた背景中の大きな欠損部分の点群データを平面近似により補完し、点群データを修復する手法を提案する。

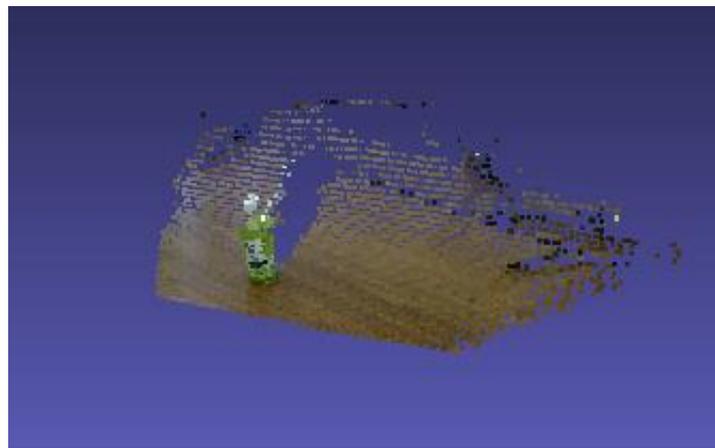


図 1 物体によって欠損ができていない点群データ

## 2. 点群修復

### 2.1 点群修復とは

点群修復とは、点群データの欠損部分の色と形状を周りの点群データの情報  
を参照して自動的に補完する技術のことである。点群データの色と形状はそれ  
ぞれ RGB 値と Depth 値によって表す。

### 2.2 点群修復の従来研究

点群データの欠損を修復する点群修復の研究は画像修復に比べて事例が少な  
いがいくつか行われている。例えば、中尾ら[1]は点群データにプリミティブ  
当てはめを行い、デプスマップとテクスチャ画像を利用して修復し、3次元モ  
デリングを行う手法を提案している。点群データから3次元モデリングを行う  
際、小さな欠損を修復することができていたが、背景中の大きな欠損には対応  
していなかった。

3次元スキャナで点群データを取得すると、背景に大きな欠損ができる。そ  
のため、背景中の欠損を修復するには大きな欠損を点群修復しなければならない。  
画像修復で大きな欠損を修復する手法として、パッチベース修復手法があ  
る。この手法は、欠損部分の境界と似ている部分を周りから探し出し、欠損部  
分にコピーして修復する。この手法をそのまま点群修復に適用すると、修復し  
た部分の点群が真の点群と大きく異なるという問題点が生じる（図2）。これ

は、修復する時に欠損部分の周りの点群データの **Depth** 値をそのまま使って補完しているためである。この場合、点群データを様々な角度から確認すると、点群データが正常に修復されていないため違和感を与える修復結果となる。この問題に対して Doria ら[2]は **RGB** 値のパッチベース修復とデプスマップの修復によって対処する手法を提案している。デプスマップの修復によって、**Depth** 値が真の値に近づき、違和感が少なくなった。しかし、この手法では、デプスマップが滑らかに修復されるため、3面から構成されている角など、急な角度の変化がある点群データに対してうまく修復ができていなかった。

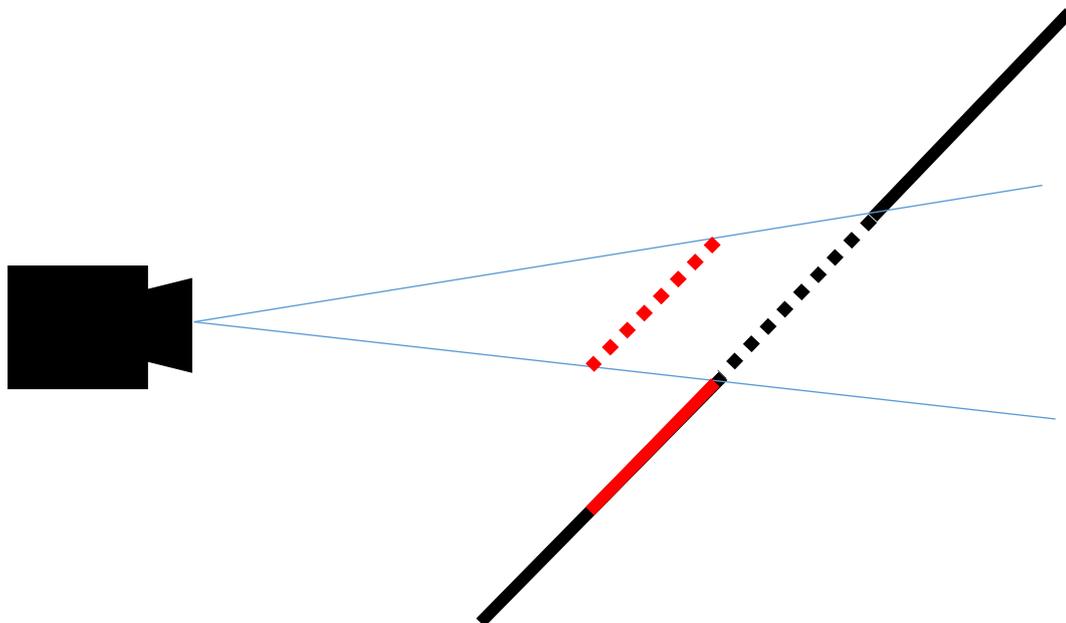


図 2 パッチベース修復における問題（黒実線：点群データ、黒点線：欠損部分、赤実線：修復時に参照された点群データ、赤点線：修復されて生成された点群データ）

## 2.3 点群修復手法

本研究では、画像修復と平面近似により点群データを修復する手法を提案する。この手法では、欠損部分が部分的に平面近似できると仮定して、3次元の点を平面上に投影し、画像を作成する。作成した画像に画像修復の技術を用いてRGB値を修復する。その後、修復した画像上の点を3次元空間に逆投影することでDepth値を修復する（図3）。この手法は、欠損部分のDepth値の真の値が平面近似可能であれば、修復後のDepth値がより真の値に近づき、違和感がなくなる。

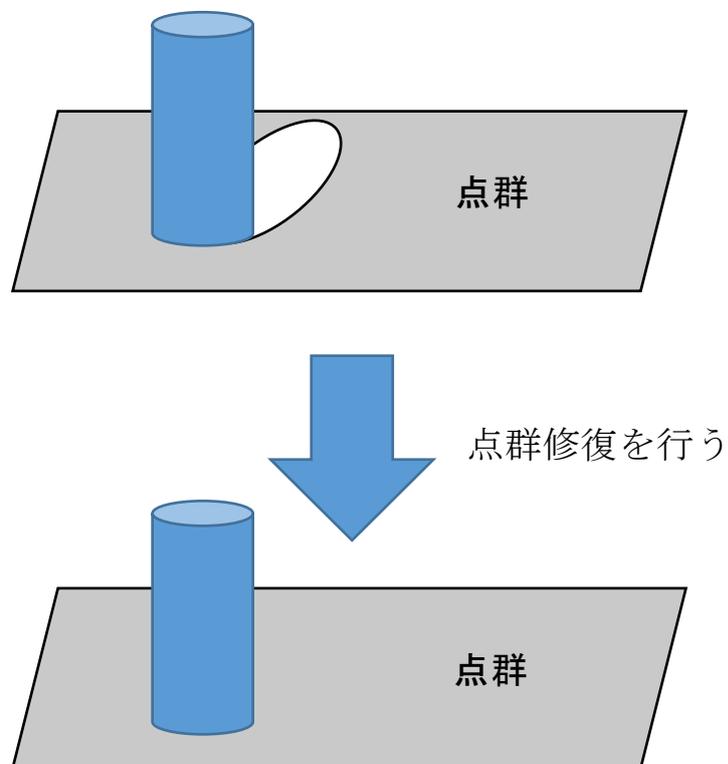


図 3 点群修復の例

### 3. 提案手法

#### 3.1 提案手法の流れ

本研究の提案手法の流れは以下の通りである。

- 1) 点群データを取得する。
- 2) 取得した点群データを平面近似により複数の平面上に分割し、それぞれの点群データを抽出する。
- 3) 抽出した点群データを画像に変換する。
- 4) 変換した画像を修復する。
- 5) 画像を点群データに逆変換する。

## 3.2 点群データの取得

本研究では、Kinect v2[3]を利用して点群データを取得する。Kinect v2 は、Microsoft 社が開発した 3 次元スキャナであり、RGB 値と Depth 値が取得できる RGBD カメラの一種である（図 4）。3 次元スキャナの中では比較的安価で気軽に利用することができる。



図 4 Kinect v2  
<http://www.xbox.com/ja-JP/xbox-one/accessories/kinect> から引用

### 3.3 平面上の点群データの抽出

取得した点群データをいくつかの平面で近似し、それぞれの平面に近い点からなる点群データに分割、抽出する。そのために[4]の点群の平面近似手法を使用する。この手法は、RANSAC アルゴリズム[5]に基づいた手法である。まず、点群データから適当な 3 点の座標をランダムに抽出し、平面の式を求める。次に平面の近く（平面と点の距離が閾値内）にあるすべての点を平面上の点とする。この処理を一定回数繰り返し、点の数が最大となる平面を探し出す。求めた平面付近の点群データを平面上の点群データとして抽出する。残りの点群データに対して、同様の作業を求めたい平面の数だけ繰り返して、点群データを平面上の点群データに分割する。

### 3.4 点群データから画像への変換

平面上の点群データを抽出した後、それぞれの点群データを画像へ変換する。この変換では3次元の点を平面上に投影することで画像を作成する。

平面L上の点群データに含まれる点を $P(x, y, z)$ と表す。この点 $P(x, y, z)$ から画像座標 $Q(u, v)$ を次式で求める。

$$\begin{cases} u = (\vec{u} \cdot P - u_{min}) * k \\ v = (\vec{v} \cdot P - v_{min}) * k \end{cases} \quad (1)$$

ここで、 $\vec{u}, \vec{v}$ は平面L上の3次元単位ベクトルであり $\vec{u} \perp \vec{v}$ を満たしている。この単位ベクトルは、平面L上に任意に定めた原点 $A(x_0, y_0, z_0)$ から求めている。 $k$ は、画像を拡大するためのものである。これは、元の点群データの座標が極めて小さいため行う処理である。式(1)で求めた対応する画像座標にRGB値を格納して、画像を作成する。本研究では画像座標が $1000 \times 1000$ 以内に収まるように設定しており、点と点の間に隙間がある画像になる(図5)。

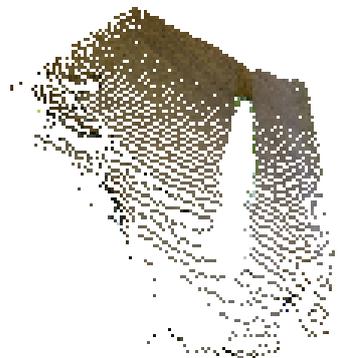


図5 点群データから変換した画像

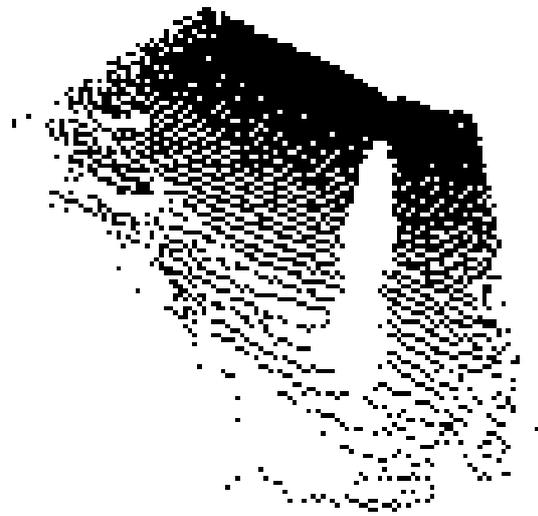
### 3.5 画像修復

本研究では Telea[6]が提案した画像修復のアルゴリズムを利用する。実装には OpenCV の INPAINT\_TELEA を利用する。画像修復では、原画像とマスク画像 (RGB 値(255,255,255)がマスクとなる画像 (図 6b)) を与える必要がある。与えた後は、マスク画像に従って修復する領域の境界から内側に向かって徐々に修復していく。

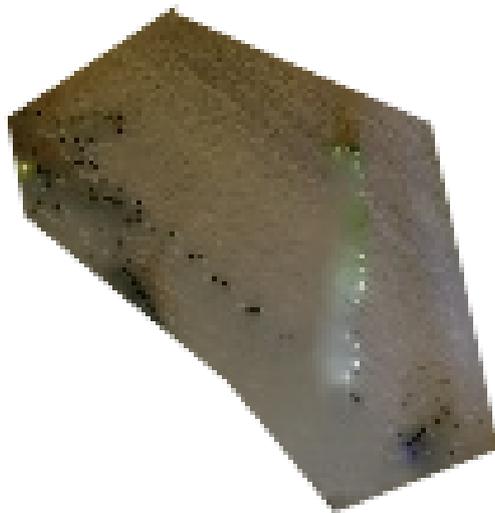
マスク画像は、点群データから画像に変換する時に、点群データが存在しない画像座標の RGB 値を(255,255,255)、存在する画像座標の RGB 値を(0, 0, 0)として作成している。また、画像修復した後の画像は点群データの取得範囲外の部分も含まれているので、その部分は手作業で切り取っている (図 6c)。



a)原画像



b)マスク画像



c)修復した画像

図 6 画像修復の実行結果

### 3.6 画像から点群データへの変換

画像修復した画像をもとの点群データに戻すには、まず次式により、画像座標  $Q(u, v)$  を 3次元座標  $P'$  に変換する。

$$\begin{cases} x_a = \frac{u}{k} + u_{min} \\ y_a = \frac{v}{k} + v_{min} \end{cases} \quad (3)$$

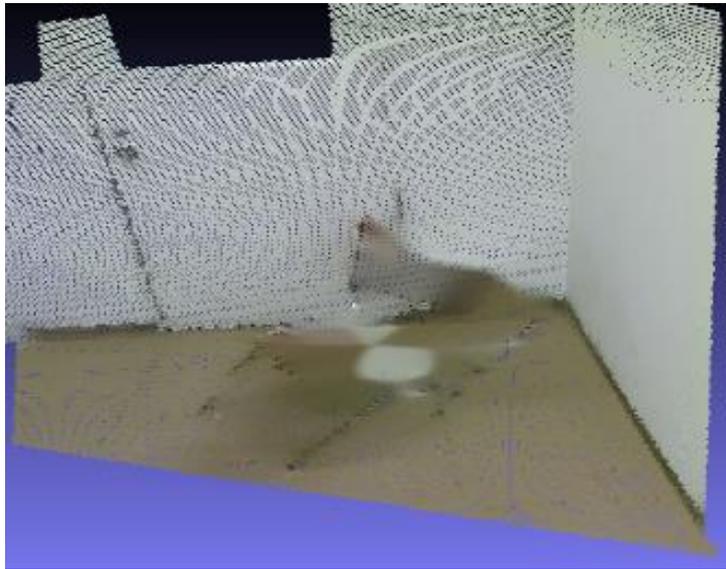
$$P' = A + x_a \cdot \vec{u} + y_a \cdot \vec{v} \quad (4)$$

式(3)では、点群データから画像へ変換したときに施した拡大・平行移動をもとに戻すために縮小・平行移動をさせている。式(3)(4)の順番で求めることで画像座標  $Q$  から 3次元座標  $P'$  へ変換することができる。

平面近似により点群を複数の平面上に分割しているため、元の点群データと抽出された点群データには差がある。この差はわずかなものであるため、確認しても違和感がない。しかし、この差のために、抽出された平面上の点群データ同士の境界が一致しないという問題が生じる (図 7)。一致していない境界を重なり合うように、対応する境界の端点同士を与え平行移動により変換する。これによって、境界の不一致の問題に対処する。



a) 境界が一致していない点群データ（赤い枠内の境界）



b) 変換後の点群データ

図 7 境界が一致していない点群データの例

## 4. 点群修復実験

### 4.1 実験手順

実験は以下の2つの点群データで行う。

- 1) 大きな欠損がある点群データ
- 2) 3つの平面から構成されている角の点群データ

評価として、欠損部分を修復して得られた点群データを確認し、どの角度から見ても違和感がないか調べる。

実験に使用する点群データにノイズが入っていた場合、実験でその影響を受ける可能性がある。そのため、できる限りノイズを減らすように、点群データを取得する際には、カメラのブレが起きないように Kinect v2 を三脚に固定してスキャンを行う。

## 4.2 大きな欠損がある点群データ

### 4.2.1 使用する点群データ

机の上に物体を置いてスキャンした点群データを使用する（図 8）。点群データから抽出する平面の数は 1 つとして実験を行った。この実験では、大きな欠損を修復できるかどうかを確認する。

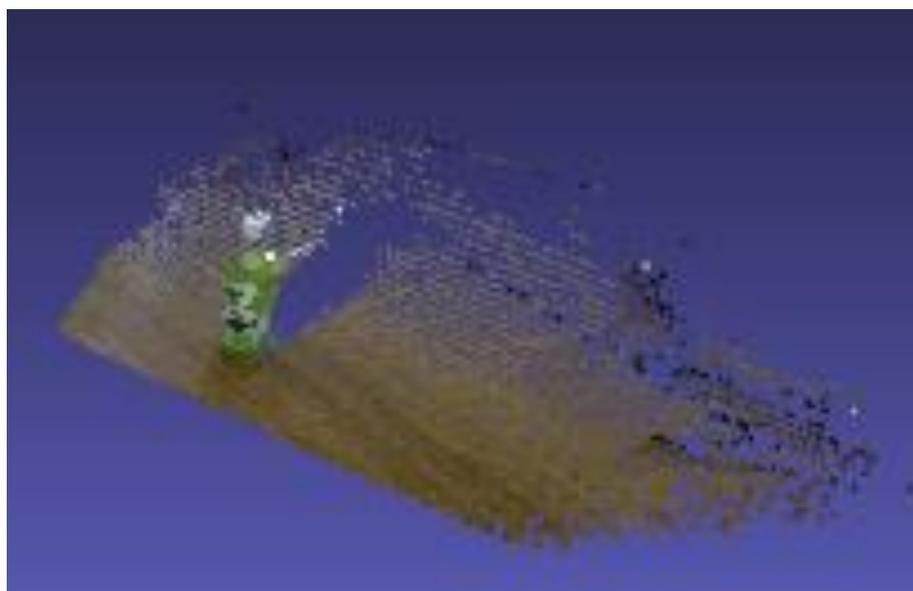


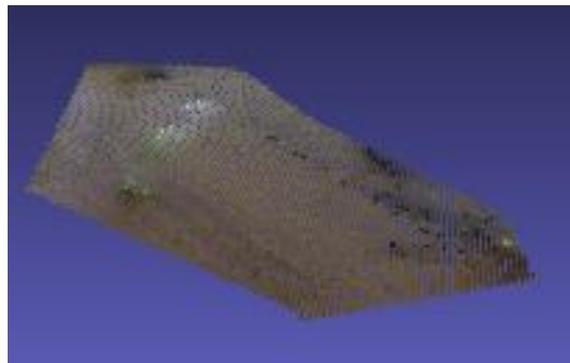
図 8 使用した点群データ

## 4.2.2 実験結果

取得した点群データと本手法による点群修復の結果を図 9、10 に示す。この実験では、修正した部分をどの角度から見ても **Depth** 値には違和感がなかった。しかし、**RGB** 値に関しては机の上の物体の影響が残っており、違和感を与える結果となった。

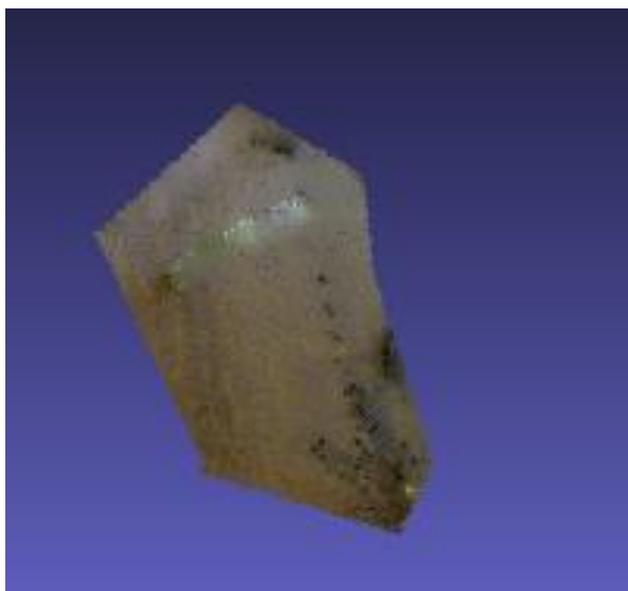


a)修復した点群データ



b)修復した平面上の点群データ

図 9 大きな欠損がある点群データの実験結果(1/2)



2か所から見た修復した平面上の点群データ

図 10 大きな欠損部分がある点群データの実験結果

## 4.3 3つの平面から構成される角の点群データ

### 4.3.1 使用する点群データ

3つの平面から構成されている角に椅子を置いてスキャンした点群データを使用する（図 11）。点群データから抽出する平面の数は3つとして実験を行った。

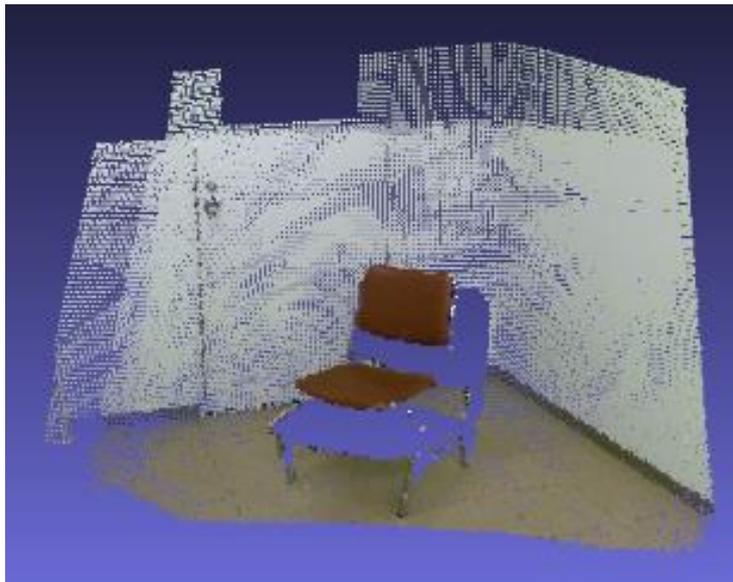


図 11 使用した点群データ

### 4.3.2 実験結果

取得した点群データと本手法による点群修復の結果を図 12、13 に示す。修復した点群データは、角の欠損部分がどの角度から見ても Depth 値には違和感がなかった。

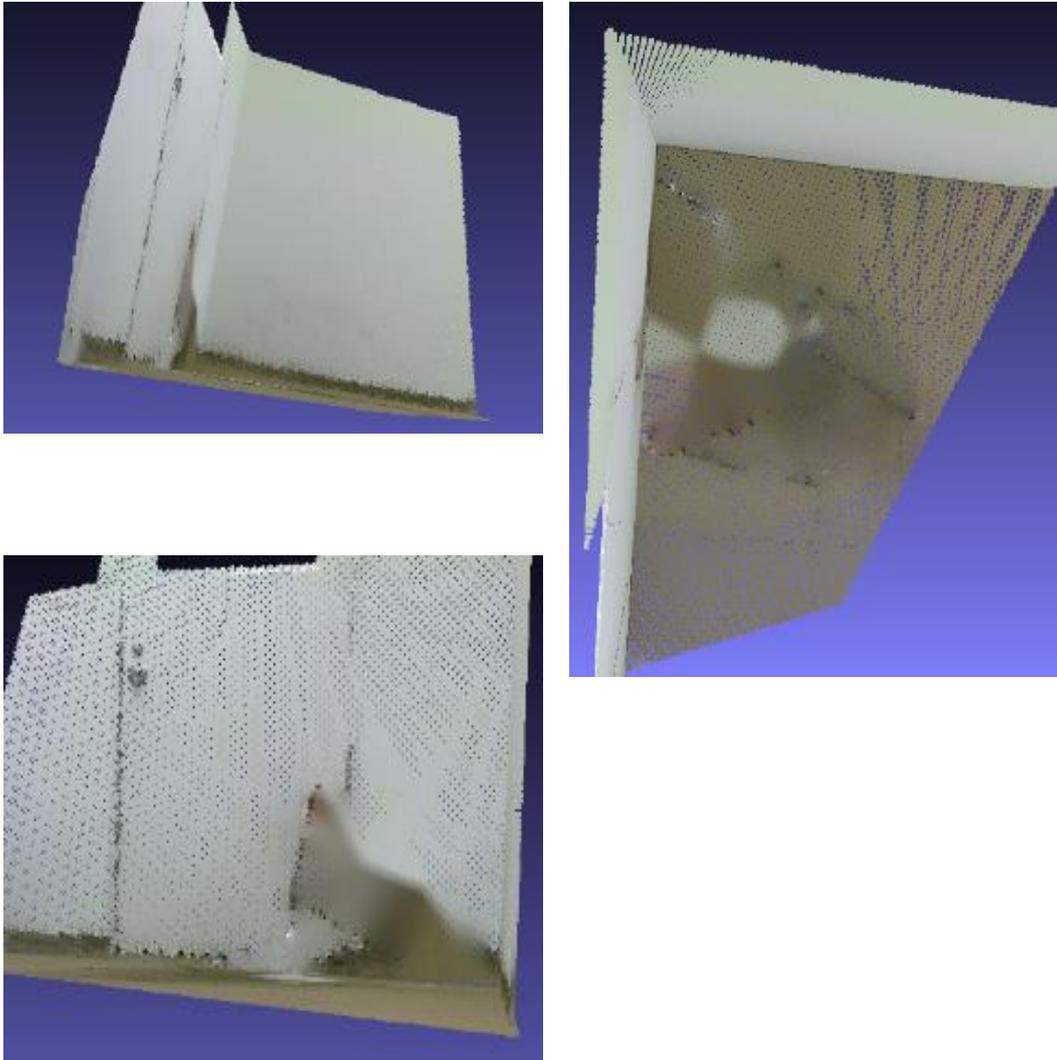


a)修復した点群データ



b)修復した平面上の点群データ

図 12 3つの平面から構成されている角の点群データの実験結果(1/2)



3 か所から見た修復した平面上の点群データの側面図

図 13 3つの平面から構成されている角の点群データの実験結果(2/2)

## 5. おわりに

本研究では、点群データを平面近似して画像に変換し、欠損部分を修復した。これにより、修復した点群の部分の色や形状が真の点群と異なる問題を改善した。

この手法は修復する点群が平面上であるという条件が前提となっている。そのため、平面でない複雑な点群データ（曲面など）ではうまく修復できない。また、本研究では Depth 値の修復に注目して実験を行ったが、画像修復では RGB 値がうまく修復されず、欠損部分に違和感が生じていた。今後、複雑な面での修復手法と画像修復の精度向上を模索していきたい。

## 謝辞

最後に、本研究を行うにあたり、指導教員である椋木雅之教授には、お忙しい中適切なご指導とアドバイスをして頂き大変感謝しております。また、椋木研究室の皆様、お忙しい中数々の助言やアドバイスをして頂きありがとうございました。これまでご指導、協力していただいた皆様に心よりお礼申し上げます。

## 参考文献

[1]中尾 聡志, 河合 紀彦, 佐藤 智和, 横矢 直和 “点群へのプリミティブ当てはめに基づく形状とテクスチャの欠損修復”, 信学技報 PRMU, Vol113, No432,pp.165-170 (2014)

[2]D. Doria, R. J. Radke, “Filling Large Holes in LiDAR Data By Inpainting Depth Gradients”, Proc. CVPR Workshops, pp.65-72 (2012-06)

[3]Kinect v2,

<https://developer.microsoft.com/en-us/windows/kinect>

[4]二宮 龍之介 “平面で構成された実物体の3次元モデリングと整形”, 宮崎大学工学部情報システム工学科卒業論文 (2016)

[5]M. A. FISCHLER, R. C. Bolles, "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography", Comm. ACM, Vol.24, No.6, pp.346-359 (1981)

[6] A.Telea, “An Image Inpainting Technique Based on the Fast Marching Method”. Journal of Graphics, GPU, and Game Tools Vol9, No1, pp 23-34 (2004)