Vol.2014-CVIM-192 No.22
2014/5/15

Kinectから取得される座標データの補正と物体の体積推定の試み

齊藤 圭佑[†1,a)]　宮城 茂幸[1]

概要：Kinect からは RGB 画像の各画素に対応した座標データ (X, Y, Z) を取得することができるが，これらのデータは必ずしも正確でない．そこで，Kinect に対し平行に置かれた平板上の既知のドットパターンを用い座標 (X, Y, Z) の補正を行った．Z 座標については，平行平板の設置位置を変えることにより，(X, Y) 座標については，ドット間の距離を調べることにより誤差の評価を行った．その結果，Z 方向については測定距離に応じて指数関数的に測定誤差が，また，(X, Y) 方向については X-Y 平面内で一定の割合で誤差が発生することがわかった．これらの誤差を打ち消すように補正を行った．1 つの応用例として，補正後の座標データを用い，単純なカービング法を用いて物体の 3 次元形状を再構成し，その体積推定を行い，推定精度を評価した．

1. はじめに

近年自動車や家電，携帯電話などの場面で様々なセンサが使用されている．その中に，センサから物体までの距離を計測する距離センサがある．距離センサの利用される場面として，行動監視，ロボットの位置制御，ジェスチャー認識，3 次元モデルの作成などが挙げられる．特に 3 次元モデルの作成では，形状認識や体積計算などの応用が考えられる．例えば，医療分野に着目すると，浮腫患者の患部の体積測定などに距離センサを利用できれば，非接触での測定が可能であり，人手による測定よりも，より正確な測定が可能になると考えられる．

絶対位置の推定が可能な距離センサとして，レーザーレンジスキャナやステレオカメラ，深度カメラなどが存在する．レーザーレンジスキャナは高密度計測が可能であるが, 計測に時間がかかる．ステレオカメラは 2 つのカメラを使用するため，補正が非常に複雑である. さらに, TOF(Time of Flight) 方式の深度カメラもあるが，これらの距離センサは非常に高価である．これに対して，Microsoft 社により開発された投影方式の深度カメラである Kinect は，主にコンピュータゲーム用に開発されたものであるため，他の距離センサに比べて安価である．

[1] 滋賀県立大学
The University of Shiga Prefecture
[†1] 現在，滋賀県立大学大学院
Presently with Graduate school of The University of Shiga Prefecture
[a)] oz23ksaitou@ec.usp.ac.jp

また, Kinect のソフトウェア開発キット (Kinect for Windows SDK) によって提供される関数を利用することで，Kinect から実際の座標データを取得することができる．ただし，取得される座標データは必ずしも正確な値ではない．

本稿では，Kinect より取得される座標データを検証し，その補正を行う．また，1 つの応用先として，単純なカービング法を用いて物体の体積推定を試みる．対象物体を正面および背面の 2 方向から Kinect により撮影し，取得された座標データから物体体積推定を行い，推定結果を評価する．

2. Kinectによる座標データの取得

本節では Kinect により物体の座標データを取得する方法を述べる．

2.1 Kinect の概要

本研究では，Microsoft 社の Kinect for Windows を使用する．**図 1** に外観図を示す．Kinect は RGB カメラ，近赤外線カメラ，プロジェクタで構成され，RGB 画像および深度画像が出力される．

文献 [1] によると，Kinect の深度測定の原理は以下の通りである．**図 2** において，点 O が近赤外線カメラの中心であるとし，そこから b 離れた位置にプロジェクタがあるとする．既知の距離 Z_o 離れた参照面に対し，既知のパターンを投影すると，その点 Q における反射はカメラの中心から f 離れた投影面上では点 y に投影される．

次に，距離 Z_k 離れた物体面上に既知のパターンを投影

図 1　Kinect for Windows の外観

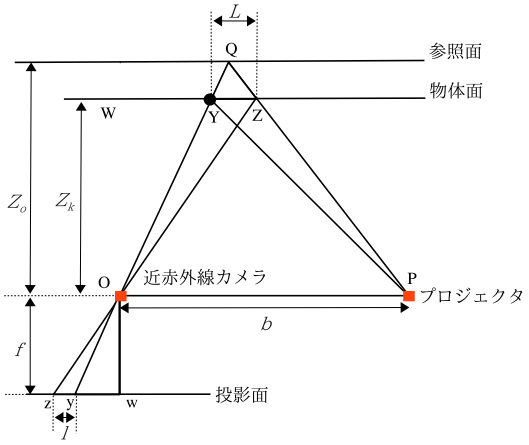


図 2　深度測定原理

し，その物体面からの反射結果を測定する．このとき，その結果は投影面上では点 z に投影される．

参照面からの反射パターンと，物体面からの反射パターンの相関を計算すると，投影面上でのずれ l が算出できる．前述の参照面からの投影結果 y は物体面上では点 Y からの反射とみなせるので，三角形 QOP と QYZ の相似より，

$$\frac{L}{b} = \frac{Z_o - Z_k}{Z_o} \tag{1}$$

となり，また，三角形 OYZ と Oyz の相似より，

$$\frac{l}{f} = \frac{L}{Z_k} \tag{2}$$

が得られる．ここで，式 (2) から式 (1) へ L を代入すると次式が得られ，物体までの深度 Z_k を求めることができる．

$$Z_k = \frac{Z_o}{1 + \frac{Z_o}{fb} l} \tag{3}$$

2.2　座標系

測定物体と Kinect のカメラとの位置関係を確認する．**図 3** のように，カメラ中心を原点として実空間座標系 XYZ をとる．カメラ中心から f 離れた垂直な面内にカメラ座標 xy をとる．なお，スクリーン座標としてはカメラから見て画像の左上角を原点とすることが多いが，Kinect では画像右上角を原点とする．これより，スクリーン座標上での行 (本稿の図では row と示す) と，列 (本稿の図では col と示す) は x, y 方向とは逆となる．

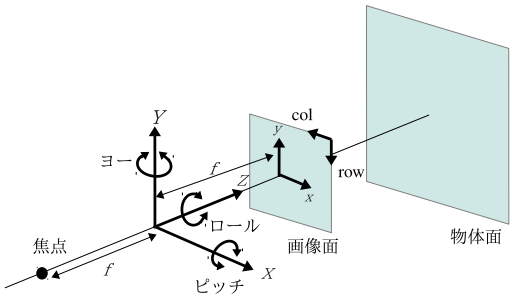


図 3　座標系

2.3　座標データの取得

図 2 からもわかるように，あらかじめキャリブレーションを行い焦点距離を求めておけば，得られた深度値と投影面上の位置から 3 次元空間内における座標 X，Y を求めることができる．Kinect for Windows SDK[2] ではこの手順を行う関数 NuiTransformDepthImageToSkeleton が提供されている．本稿では，NuiTransformDepthImageToSkeleton 関数を利用して，Kinect から座標データを取得する．ここで，Microsoft Developer Network のホームページ [3] によれば，NuiTransformDepthImageToSkeleton 関数が出力する座標データの単位は m(メートル) とされている．しかし，予備実験により関数の出力を確認したところ関数により得られる値は実測値の 1/10 倍となることがわかった．そこで，本稿では NuiTransformDepthImageToSkeleton 関数によって得られる座標データを 1/10 倍し，以降は単位を mm に換算して表記する．

また，Kinect から出力される RGB 画像と深度画像について，RGB カメラと近赤外線カメラおよびプロジェクタの位置は異なっているため，両者のスクリーン座標は一致しない．ここでは簡単のため，Kinect for Windows SDK が提供する NuiImageGetColorPixelCoordinatesFromDepthPixelAtResolution 関数を用いて，深度画像のスクリーン座標を RGB 画像のスクリーン座標に一致させる．

3.　取得された座標データの補正

深度画像のスクリーン座標 (row, col) およびカメラ解像度を与えることで，NuiTransformDepthImageToSkeleton 関数より実空間における座標データ (X, Y, Z) が得られる．得られた座標データがどの程度正確か評価するとともに，補正方法について述べる．

Kinect に対して平行に平板を設置し，撮影を行う．平板上には，**図 4** に示すドットパターンが印刷されている．NuiTransformDepthImageToSkeleton 関数より得られた座標データ (X, Y, Z) を各座標ごとにカメラ解像度と同じ大きさの画像形式の行列として表現する．以下，これらの行列をそれぞれ X 画像，Y 画像，Z 画像と呼ぶこととする．

Kinect の起動直後は，取得される深度が安定しないことがある．したがって Kinect を起動後，5 分間経過してから

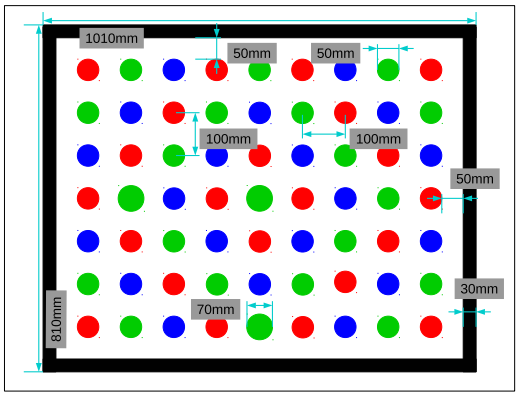


図 4　ドットパターン

表 1　Z 座標の平均値と誤差率

Kinect-平板 [m]	Z 座標の平均値 [mm]	誤差率 [%]
0.50	590	15.232319
0.75	899	16.545255
1.00	1211	17.397817
1.25	1525	18.041270
1.50	1842	18.573491
1.75	2168	19.298798
2.00	2486	19.552987

撮影を行うようにした．また，以下の実験では測定範囲を 3m に限定しているので Kinect を Near モードに設定し，撮影を行った．

3.1　Z 座標について

Kinect-平板間の距離を 0.5m, 0.75m, 1.0m, 1.25m, 1.5m, 1.75m，2.0m に設定し，それぞれに対して 10 回ずつ測定を行う．

各距離において得られた Z 座標の平均値と誤差率を**表 1** に示す．Z 座標の平均値は，10 回の測定で得られた Z 画像について，(150-314 行，212-430 列）の矩形領域内の画素のうち欠損値，すなわち Z 座標が 0 の画素を除いた画素値の平均をそれぞれ計算し，さらに得られた 10 個の平均値の平均を計算した値である．誤差率は，平均値に対する実際の距離と平均値との差であり，次式によって計算する．

$$\text{誤差率} = \frac{Z\text{ 座標の平均値} - \text{実際の距離}}{Z\text{ 座標の平均値}} \times 100\,[\%] \quad (4)$$

図 5 に横軸を Z 座標の平均値，縦軸を誤差率の絶対値としてプロットしたグラフを示す．図 5 から誤差率は Kinect-平板間の距離が大きくなるにつれて増加するが，誤差率の増加は次第に小さくなっていくことがわかる．また横軸を対数軸に変更すると**図 6** となり，ほぼ線形な変化が確認できる．そこで，対数プロットにおいて最小自乗法による線形近似を行い，平均値と誤差率の関係を対数関数として近似する．近似式は式 (5) のようになる．ただし，式 (5) における Z 座標値の単位は m/10 である．図 5 の曲線および図 6 の直線は近似式を示している．

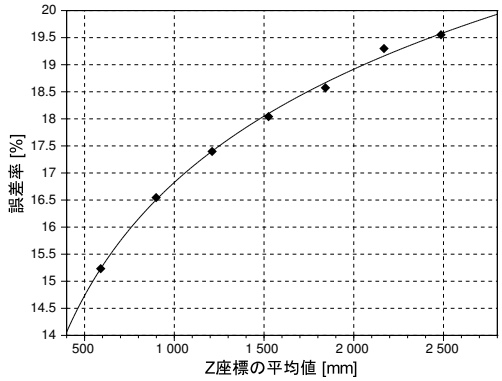


図 5　Z 座標の平均値と誤差率の関係（線形プロット）

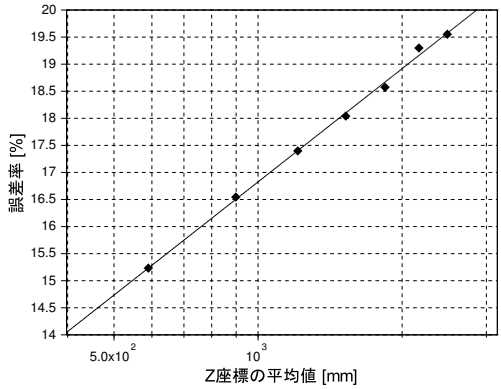


図 6　Z 座標の平均値と誤差率の関係（片対数プロット）

$$R(Z) = 3.0171590 \times \ln Z + 23.773020\,[\%] \quad (5)$$

この結果より，Z 座標の補正は，式 (5) を用いて誤差率を計算し，得られた誤差率より Z 座標に対する誤差量を求め，これを Z 座標から減算することで行うこととする．

3.2　X および Y 座標について

RGB 画像として撮影されたドットパターンを利用して，取得された X および Y 座標の精度を検証する．RGB 画像に対して 2 値化処理を行い，各ドット領域の重心画素位置を決定する．パターン中央のドットを基準とし，**図 7** に示すドット間距離 D_1, D_2, D_3, D_4 を推定する．

表 2，**表 3**，**表 4** に Kinect-平板間が 1.0m，1.5m，2.0m におけるドット間距離の推定結果を示す．表 2，表 3，表 4 より，どの Kinect-平板間距離においても，ドット間距離の推定結果は実際の距離よりも大きくなることがわかる．また，**表 5**，**表 6** に列方向および行方向の各ドット間の距離の平均値，最大値，最小値を示す．各ドット間距離には一定でない誤差が認められるが，これは RGB カメラの解像度の関係で 2 値化されたドット領域の形が完全な円形にならないことにより，重心画素位置に誤差が生じたことが理由として考えられる．この結果より，X 座標については係数 $a = 0.8484986$，Y 座標については係数 $b = 0.8523856$

表 2　距離推定結果 (Kinect-平板 1.0m)

対象	実際の距離 [mm]	推定結果 [mm]
D_1	100	116
D_2	200	235
D_3	100	116
D_4	200	233

表 3　距離推定結果 (Kinect-平板 1.5m)

対象	実際の距離 [mm]	推定結果 [mm]
D_1	100	116
D_2	200	238
D_3	100	119
D_4	200	236

表 4　距離推定結果 (Kinect-平板 2.0m)

対象	実際の距離 [mm]	推定結果 [mm]
D_1	100	117
D_2	200	234
D_3	100	118
D_4	200	240

表 5　列方向の各ドット間距離推定

Kinect-平板 [m]	平均 [mm]	最大 [mm]	最小 [mm]
1.0	118	122	114
1.5	118	124	113
2.0	118	124	111

表 6　行方向の各ドット間距離推定

Kinect-平板 [m]	平均 [mm]	最大 [mm]	最小 [mm]
1.0	117	122	114
1.5	118	122	116
2.0	118	123	115

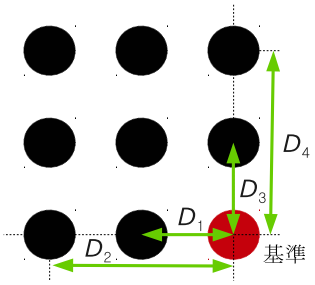


図 7　距離推定対象

を乗じることで補正を行うこととする．係数 a, b は式 (6) および式 (7) により計算する．係数 a, b は，Kinect-平板間が 1.0m のデータにおいて求めた．

$$a = \frac{\text{実際の距離}}{\text{列方向の各ドット間距離推定値の平均値}} \quad (6)$$

$$b = \frac{\text{実際の距離}}{\text{行方向の各ドット間距離推定値の平均値}} \quad (7)$$

4. カービングによる物体の体積推定

本節では，Kinect により取得された座標データから測定対象物体の体積を直接推定する方法を述べる．深度画像を利用した様々な 3 次元形状復元法が提案されている．例えば Curless は複数の深度画像を重み付けして統合することにより物体の表面形状を復元する手法を提案している [4]．このような手法を用い，物体表面の形状を一旦復元した後，体積を推定する方法が考えられる．しかし，復元のためのコスト計算が大きくなるため，従来のカービング手法 [5] をボクセル集合に適用し，2 方向から撮影された深度画像から直接的に体積推定を行った．

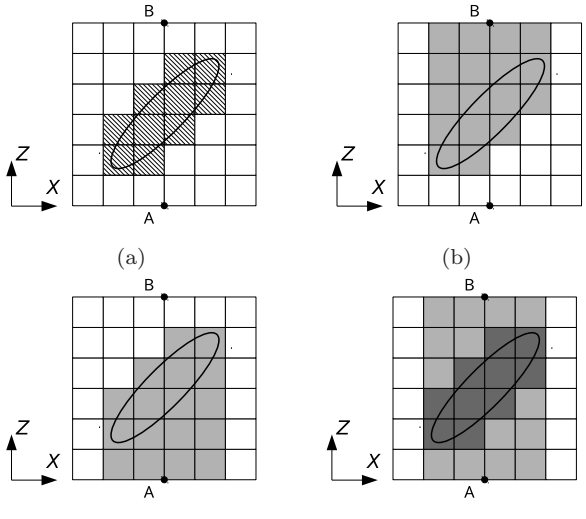


図 8　カービング法を用いた体積推定の原理．(a): 測定対象を含むようにボクセルの集合を設定する．(b): 視点 A より撮影した深度画像よりカービングを適用し，残ったボクセル．(c): 視点 B より撮影した深度画像よりカービングを適用し，残ったボクセル．(d): (a)，(b) を統合し最終的に残ったボクセル

深度画像を用いたカービングの原理を図 8 に示す．簡単のため図では 2 次元表示を行っている．図 8(a) においてだ円部分が測定対象物体であるとすると，その測定対象を十分含むボクセル集合を設定する．図 8(b) では，A 点から Z 軸方向へ測定した深度画像からボクセルを削り，残ったボクセルを表している．同様に図 8(c) では，B 点から Z 軸の負の方向へ測定した深度画像からボクセルを削り，残ったボクセルを表している．これらの結果を統合し，最終的に残ったボクセルを図 8(d) に示す．これらのボクセルが測定対象物体の形状を表しているので，ボクセルの総数を数えることにより対象物体の体積を推定できる．後の評価実験ではボクセルの大きさを 1cm × 1cm × 1cm とした．

5. 評価実験

本節では，Kinect より取得された座標データの補正結果，2 方向から撮影されたデータを用いた物体体積推定の実験結果について述べる．

5.1 Z 座標の補正

平行平板を撮影して作成した Z 画像について補正を行

表 7　補正前後における Z 座標と誤差

	Kinect-平板 [m]	Z 座標の平均値 [mm]	最大誤差 [mm]
補正前	0.50	590	90
	0.75	899	150
	1.00	1210	220
	1.25	1525	290
	1.50	1842	370
	1.75	2169	440
	2.00	2485	540
補正後	0.50	500	8
	0.75	751	6
	1.00	1000	9
	1.25	1249	13
	1.50	1498	20
	1.75	1753	27
	2.00	1998	41

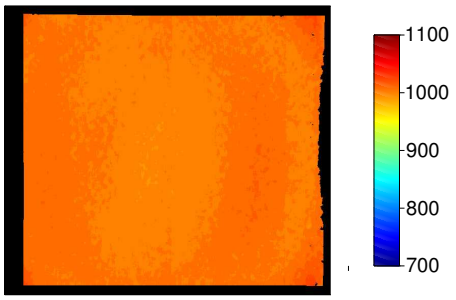


図 9　Z 画像のカラーマップ（Kinect-平板 1.0m）

う．補正を行う対象として，異なる Kinect-平板間距離について 10 回ずつ撮影したデータのうち，2 番目のデータを使用する．

Kinect より取得される Z 座標は，実際の距離よりも大きい値であった．**表 7** より，補正後の Z 座標値は実際の距離に近い値となることがわかる．

図 9 に Kinect-平板が 1m の場合の補正後の Z 画像のカラーマップを示す．図 9 より，画像中央の部分と比較して，画像の四隅では Z 座標の値が大きく取得されていることがわかる．一般に画像の周辺部分ではレンズによる歪みが発生しやすいため，不正確な値が計測されると考えられる．そこで，体積推定の評価実験においては，有効な測定範囲として 50-400 行，150-500 行の矩形領域を設定し，この領域内で取得される座標データのみを利用することとする．

5.2　X および Y 座標の補正

平行平板を撮影して作成した X 画像および Y 画像について補正を行う．補正の対象は，Kinect-壁面が 1.0m, 1.5m, 2.0m のデータとし，補正後のデータとドットパターンを用い，3.2 と同様にしてドット間距離の推定を行う．

表 8　補正前後のドット間距離の推定結果 (Kinect-平板 1.0m)

	対象	実際の距離 [mm]	推定結果 [mm]
補正前	D_1	100	116
	D_2	200	233
	D_3	100	116
	D_4	200	233
補正後	D_1	100	99
	D_2	200	198
	D_3	100	100
	D_4	200	199

表 9　補正前後のドット間距離の推定結果 (Kinect-平板 1.5m)

	対象	実際の距離 [mm]	推定結果 [mm]
補正前	D_1	100	116
	D_2	200	236
	D_3	100	119
	D_4	200	240
補正後	D_1	100	98
	D_2	200	201
	D_3	100	102
	D_4	200	204

表 10　補正前後のドット間距離の推定結果 (Kinect-平板 2.0m)

	対象	実際の距離 [mm]	推定結果 [mm]
補正前	D_1	100	117
	D_2	200	234
	D_3	100	117
	D_4	200	240
補正後	D_1	100	99
	D_2	200	199
	D_3	100	100
	D_4	200	204

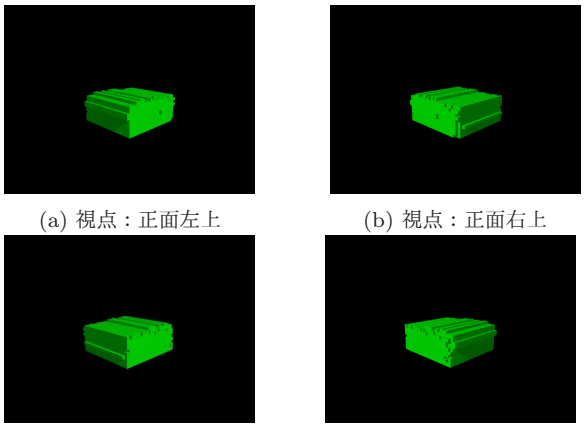
表 8, **表 9**, **表 10** より，正後の推定結果は，どの推定対象でも実際の距離に近い値になることがわかる．

5.3　物体形状および体積の推定

4 節で述べた方法を用いて，物体形状および体積の推定を行う．撮影する対象として直方体の箱を使用する．箱の大きさは，25cm × 26cm × 26cm である．この箱を正面および背面の 2 方向から撮影したデータにより推定を行う．想定するボクセル集合の要素数は 200 × 200 × 200 とする．これが，1 辺が 2m である立方体の実空間に対応する．

今回の実験は，1 台の Kinect を使用して行う．2 方向からのカービングでは，正面および背面から撮影を行う位置を正確に決めておく必要がある．この場合，前面からの撮影のあとに背面の撮影位置に改めて Kinect を設置する必要があり，正確に設置することは難しいと考えた．そこで，ボクセル集合全体をその中心を基準として 180 度回転させることで，これを背面からの撮影データとすることを考える．今回は対象物体の中心をボクセル集合の中心と一致させて，対象物体を 180 度回転させることで擬似的にこれを

図 10　正面撮影時の RGB 画像



(a) 視点：正面左上　(b) 視点：正面右上

(c) 視点：背面左上　(d) 視点：背面右上

図 11　カービングにより残ったボクセル

実現する．

今回の実験では，有効範囲内の座標データのみを利用するため，箱の全体を推定できない．よって，**図 10** に示す RGB 画像の有効範囲内において，箱の映っている部分をあらかじめ求めておいた．本実験で推定する部分の大きさは 25cm × 10cm × 26cm である．

図 11 にカービングにより残ったボクセルを示す．また，残ったボクセルについて，その要素数を数えることで体積の推定を行う．体積の推定結果は 6201cm^3 となった．推定対象となる部分の実際の体積は 6500cm^3 であり，推定結果は実際の体積に比べて小さくなった．この結果について考察する．

1 つ目の原因として，Kinect から取得される座標データにはランダムな誤差が存在するため，これが推定結果に影響した可能性がある．

2 つ目の原因として，Kinect および対象物体の設置の際に誤差があったことが考えられる．物体を 180 度回転させて背面からのデータを取得する本手法では，物体の中心は想定するボクセル集合において正確に中央に位置していなければならない．設置の際に誤差が存在すると，正面データと背面データの統合の際にずれが生じるため，体積の推定結果に影響すると考えられる．

6. おわりに

Kinect により，取得される座標データ (X, Y, Z) の補正を行った．Z 座標値に関して，補正前では最大誤差は，Kinect-平板間の距離によっておよそ 18∼27%であったが，補正後では最大誤差は，およそ 0.8∼2.1%に減少した．さらに，X および Y 座標に関して，ドットパターンにおいて既知のドット間の距離を推定すると，補正前では誤差はおよそ 16∼20%であったが，補正後ではおよそ 1∼2%となった．また，Kinect により物体を 2 方向から撮影して取得された座標データから，物体の体積推定を試みたところ，推定された体積は実際の体積に対し，およそ −4.6%となった．物体の体積推定では，1 台の Kinect を用いて，対象物体を 180 度回転させることにより，2 方向からの撮影を行ったが，Kinect と物体の設置を正確に行うことは難しく，今後は複数台の Kinect を用いた多視点からの撮影や，座標データを点群として扱い処理することを検討していきたい．

参考文献

[1] Kourosh Khoshelham and Sander Oude Elberink: *Accuracy and Resolution of Kinect Depth Data for Indoor Mapping Applications*, Sensers, Vol. 12, No. 2, pp. 1437–1445 (2012).
[2] Kinect for Windows SDK, 入手先 ⟨http://www.microsoft.com/en-us/kinectforwindows⟩ (2014.04.14).
[3] Microsoft Developer Network: Kinect for Windows SDK，入手先 ⟨http://msdn.microsoft.com⟩ (2014.04.14).
[4] Curless B. and Levoy M.: *Volumetric method for building complex models from range images*, Proc. of the ACM SIGGRAPH Conference on Computer Graphics, pp. 303–312 (1996).
[5] Kutulakos K. N. and Seitz S. M.: *Theory of shape by space carving*, International Journal of Computer Vision, Vol. 38, No. 3, pp. 199–218 (2000).