複数のRGBDカメラによるボリュームモデルのリアルタイム生成

Matsumoto, Erisa / 松本, 依里紗

(出版者 / Publisher)
法政大学大学院情報科学研究科
(雑誌名 / Journal or Publication Title)
法政大学大学院紀要. 情報科学研究科編
(巻 / Volume)
13
(開始ページ / Start Page)
1
(終了ページ / End Page)
6
(発行年 / Year)
2017-03-31
(URL)
https://doi.org/10.15002/00021533

複数の RGBD カメラによるボリュームモデルのリアルタイム生成
Real time volume modeling with multiple RGBD cameras

松本　依里紗
Erisa Matsumoto
法政大学大学院情報科学研究科情報科学専攻
E-mail: 16t0019@cis.k.hosei.ac.jp

Abstract

With the head mount displays are getting popular, virtual reality technology is put to the practical use. Many devices and applications are introduced to public markets, and consumers are able to use them readily. Most applications, however, use the shape models generated inside the computer, but ones from real objects in real world will become common in near future. It is necessary to develop a technique to generate a shape model from three-dimensional scan of a moving object, this paper proposes a method to create a three-dimensional model in real time using multiple RGBD cameras. The proposed method estimates the shape model using voxels. Simultaneously scan the entire circumference of the subject with multiple cameras. Using depth data obtained from each camera, that express the distance to the surface, we give weights to each voxels at the measured distance. The polygons which are rendered for each voxel are calculated using the Marching Cubes method. In order to reduce the cost of Marching Cube, we introduced Octree to reduce the searching area of rendered polygons. In the existing work using the surface model, the processing speed decreases with the number of cameras, but our proposed method is hardly affected. It gains four times faster than the existing method from the experiment using four RGBD cameras.

1. 序論

実世界の物体の三次元復元は，映画やゲームといったエンタテインメントコンテンツだけでなく，医療や文化遺産の保護，研究など幅広く利用されておるが，テレイマージョンのようなリアルタイムアプリケーションにも用いることができる．テレイマージョンとは，遠隔地にいる人物と体の動きなどの情報を共有することにより，テレビ電話のような画面越しでの対話ではなく，対話相手と自分自身が同じ空間に存在しているかのように認知させる技術であり，高臨場感通信とも呼ばれている．従来のテレイマージョンは，実世界に用意したスクリーンに対話相手がいる空間の映像を映し出すことで，自分自身が相手と同じ空間にいるように感じさせてたが，バーチャルリアリティを用いることで，対話相手も自分自身も仮想空間という同じ空間に存在しているように感じることができるようになる．また，機材も多方にスクリーンを配置する必要がなく，ヘッドマウントディスプレイのみで実現可能となり，一般ユーザでも扱うことができる．このような仮想空間上でのテレイマージョンを実現するには，対話するユーザを三次元アバターとして仮想空間に投影する必要がある．そのためには，現実世界の人物の三次元モデルをリアルタイムに生成しなくてはならない．そこで本研究では，テレイマージョンを仮想空間上で実現するために，現実世界の物体のリアルタイム三次元モデル生成手法の提案を行う．

Microsoft が発売した RGBD カメラの Kinect は奥行き方向を含めた実空間の情報を取得でき，安価に入手できることもあり，多くの研究に用いられてきた．この Kinect を用いた Kinect Fusion では，スキャンしたい物体を Kinect で撮影することで三次元スキャンすることができる．また，スキャン状況をリアルタイムに確認できることから，CG の専門知識のない一般ユーザでも容易に扱うことができる．しかし，1 つの物体のスキャンには Kinect を動かして全周をスキャンする必要があり時間がかかるため，対象物体が静止物体に限られるという問題がある．したがって，人間のような動体のスキャニングを行う際に僅かでも動いてしまうと三次元モデル生成が失敗してしまう．これに対して，複数の RGBD カメラを用いて被写体の全周を瞬時に取得して動体のリアルタイムスキャニングを行う手法がある．先行研究では，1 台のカメラから取得した深度データから 1 枚のモデルを生成，それをカメラ台数分重ね合わせることで全周の三次元モデルとするサーフェスモデルを用いている．しかし，サーフェスモデルではカメラの台数が増えるにつれて処理量が増加してしまう．

そこで本研究では，ボリュームベースのリアルタイム三次元モデル生成手法を提案する．ボリュームモデルでは，撮影空間をグリッド上のボリュームボクセルと考えて，各カメラから得た深度データを用いてボクセルの重みを更新する．ボクセル単位でポリゴンを求めることで三次元モデルとなる．深度データは重みの更新にのみ用いられるため，カメラ台数による速度の低下はわずかとなる．また，GPGPU による高速データ処理により更なる処理速度の向上を目指す．

以下，2 節では Kinect を用いた三次元モデル生成についての関連研究を紹介する．3 節では，関連研究をベースに作成したサーフェスベースのモデル生成手法について述べ，問題点を示す．4 節では本研究の設計，5 節では

Supervisor: Prof. Toshio Hirotsu / Prof. Satoru Fujita

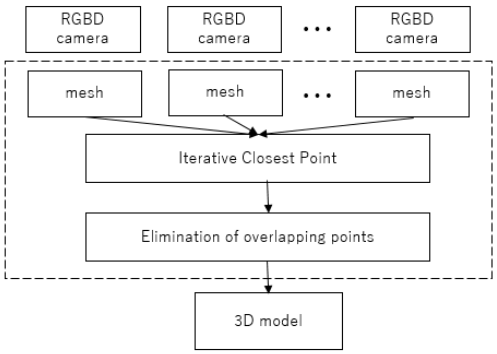

図 1　サーフェスモデルの概要

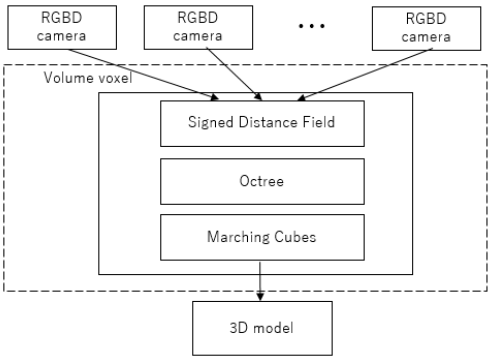

図 2　提案手法の概要

実装について述べる．6 節では提案手法の速度測定実験と結果を示し，7節では考察を行う．最後に8節で本研究のまとめを述べる．

2. 関連研究

2.1. Kinect Fusion

Izadi ら[1][2]は，リアルタイムに三次元形状の復元を行う Kinect Fusion を開発した．RGBD カメラである Microsoft の Kinect と並列演算において性能を発揮する GPGPU 技術を組み合わせ，Kinect で取得した深度マップから，物体の三次元形状を取得する手法である．スキャン方法は，物体の全周を撮影するように Kinect を連続的に動かし様々な角度，位置の深度マップを得る．その深度マップからリアルタイムに三次元モデルを描画，更新を繰り返していき，全周のスキャンが終わると 1 つの三次元モデルが完成する．

物体の全周を撮影するように Kinect を連続的に動かし様々な角度，位置の深度マップを得る．そして，空間を均一なグリッド上のボクセルボリュームと考えて，得られた深度マップの情報からボクセルデータを更新していく．ボクセルデータには近傍にある物体面との距離を多値で表現した符号付距離場が格納され，最後に Ray marching 法をボクセルボリュームに対して行い，表面推定をする．生成した三次元モデルは Kinect から深度マップを取得する度に更新される．しかし，この手法では 1 つの三次元形状を復元するために手動での全周スキャンが必要であり時間がかかる．また，全周スキャン中は被写体を動かすことができないため，人間のような動体には対応することができない．

2.2. サーフェスモデル

Alexiadis ら[3]は複数台の RGBD カメラを 4 方向に配置し，動体の三次元モデルをリアルタイムに生成する手法を提案している．深度マップは二次元画像のピクセルに対して深度値が与えられている画像である．そのため，深度マップを三次元空間の点群へ変換しても点群間の隣接関係は変わらず，隣接点同士を結び合わせることでポリゴンメッシュを生成することができる．このとき，隣接する点の深度差が閾値以上であれば三次元モデルの表面は不連続であると判断しポリゴンの生成は行わない．各カメラから生成されたメッシュを重ね合わせることにより全周の表面モデルとなるが，隣接するカメラから生成されたメッシュ間の詳細な位置合わせを行わないと三次元モデルに歪が生じてしまう．また，メッシュ間には重なる箇所が存在しており，二重に描画することは処理コストの増加に繋がる．そこで，毎フレームごとに詳細な位置合わせを行い，2 枚のメッシュ間で同一点と判定された点の一方を除去することで滑らかな表面モデルを生成する．

3. 予備実験

予備実験として，Alexiadis らの手法を参考にサーフェスベースのリアルタイム三次元モデル生成を行った．処理手順は図 1 の通りである．事前に各カメラ間のキャリブレーションを行い，各 RGBD カメラから得た深度情報をサーバ PC へ転送し三次元モデル生成の処理を行う．

事前のキャリブレーションは座標系の統一が目的であり，僅かなずれがある．そのまま三次元モデルを生成すると，2 枚の表面モデル間に僅かなずれ生じ，滑らかな 1 枚の表面モデルでなくなってしまう．そこで，各深度データ間で ICP(Iterative Closest Point)[4]による詳細な位置合わせを行う．本実験では CUDA 環境を用いるため，CUDA に対応した ICP である EM-ICP[5]を用いて実装した．位置合わせの後，隣接する点により三角形ポリゴンを形成，表面モデルとして出力する．隣接する点の深度差が閾値以上であれば三次元モデルの表面は不連続であると判断しポリゴンの生成は行わない．

2.2 節と同様に 4 台の RGBD カメラを用いて 1 フレームの処理時間を測定した結果，109msec となった．位置合わせが不要な 1 台での処理速度は 9msec/frame だったことから，詳細な位置合わせの処理コストが原因と考えられる．位置合わせは最近傍点の探索を反復するため処理時間がかかってしまう．また，複数の RGBD カメラを用いることで位置合わせ処理を複数回行う必要があり処理コストの増加に繋がる．カメラの台数が処理速度の低下の原因となると，より多くのカメラを用いる三次元復元に用いることは難しい．

4. 設計

4.1. 概要

本研究では，RGBD カメラの台数により処理速度が低下しない，ボクセルベースによるリアルタイム三次元モデルを生成し描画を行う．被写体を囲うように RGBD カメラを配置することで被写体の全周を同じタイミングでスキャンする．これにより被写体が動体でも三次元モデルを生成することが可能となる．また，提案手法では，GPGPU による処理の高速化も行う．

ボクセルベースのモデル生成では，撮影する空間を大きなボクセルとみなし，そのボクセルを細分化した小さなボクセル単位でポリゴンを構築していく．そのため，カメラの台数に関わらず三次元モデル生成の処理時間はほぼ一定となる．また，各 RGBD カメラから取得した深度データはポリゴンの描画パターンを決定する重み付けに用いるため，深度データをそのまま扱うサーフェスモデルと違い，キャリブレーションの微小な誤差の影響を受けにくい．

図 2 に提案手法の概要を示す．まず，事前処理としてカメラのキャリブレーションを行う．次に各 RGBD カメラから深度マップを取得，描画処理を行うサーバへ送信する．サーバでは，すべての深度データの情報からボクセルデータの重みを更新する．ボクセルデータの重みを用いて Marching Cubes 法により表面形状推定を行う．また，ボクセルデータを Octree 構造で保存することによりデータのない不要な空間の形状推定を省くことができる．これらの処理を GPGPU により並列に行うことで，リアルタイムでの三次元モデル生成，描画を行う．

4.1. ボクセルデータ更新

描画する空間を均一なボクセルボリュームとし，各 RGBD から取得した深度マップを用いてボクセルデータの値の更新を行う．図 3 のようにカメラから深度点までの視線上にあるボクセルのうち，深度点と近傍なボクセルに対して重み更新処理を行う．値の更新には符号付距離場(Signed Distance Fields, SDF)を用いる．符号付距離場は物体や領域の内外を符号で表したものであり，領域の内側であれば正の値，外側であれば負の値となる．また，領域の境界に近いほど符号付距離場の絶対値は小さくなる．

カメラ視点から各深度座標までの距離をd_p，ボクセルまでの距離をd_vとし，その差を符号付距離場としてボクセルの重みとする．重みが与えられるのは生成する表面モデルの近傍ボクセルのみであり，重みのないボクセルについてはデータがない空間と判定される．同時刻に取得したすべての RGBD カメラから得た深度データを用いて，ボクセルデータの重みの値を決定する．ボクセルデータの値は次のフレームには引き継がずにリセットすることで，物体の動作に影響されず毎フレームごとに異なるモデルが生成される．

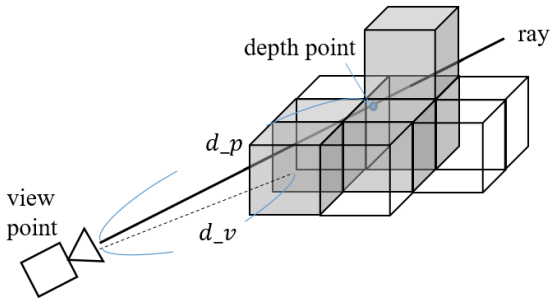

図 3　符号付距離場によるボクセルデータの更新

4.2. Octree を用いた高速化

4.3 節ではボクセルごとに処理を行うが，ボクセル数が膨大であり，計算コストがかかる．例えば，ボリュームボクセルのグリッド 1 辺の大きさが 512 であれば 1 億個以上のノードにアクセスする必要があり，リアルタイムでの処理は困難である．しかし，全てのボクセルにデータが含まれるのではく，物体の存在しない空間のボクセルはデータを保持していない．また，生成される三次元モデルの内側はポリゴンを持つ必要がないため空洞となり，同様にボクセルはデータを持たない．このような不要な空間については探索を省くことができる．

事前実験として，成人男性を被写体として，被写体から 2m の距離に RGBD カメラを 4 台配置してデータを持つボリュームボクセルの量を測定した．ボリュームボクセルの 1 辺のサイズは 2.56m であり，ボクセル数は 256×256×256 とした．結果，データを持つポリゴンを生成すべきボクセルの数は全体の 10%程度であった．残りの 90%については表面形状推定を行う必要がない．この不要な空間を探索から除去する方法として Octree 構造を用いる．Octree は子を 8 個持つ 8 分木の木構造であり，モデルを生成する空間を大きな 1 つのボクセルとみなし，それを X・Y・Z 軸それぞれの方向に 2 分割する．分割したボクセル内にデータが含まれていなければ，そのノードの探索は終了する．データが含まれていれば同様に分割し更に深く探索を続ける Octree によりデータを持つボクセルのみを取り出し，そのボクセルに対して表面形状推定処理を行う．

4.3. 表面形状推定

4.1 節で求めたボクセルデータの重みから三次元モデルを構成するポリゴンを求める．ボクセルデータの重みは被写体より外側は負，内側が正となっており，重みの符号の境界にポリゴンを構築してモデルを生成する．たがいに隣接する 8 つのボクセルデータについて考えていき，これらの 8 つの値の符号から Marching Cubes 法[6]によってポリゴン群を割り当てる．割り当てられる三角形のパターンは 256 通りであり，対称性を考慮すると 15 通りとなる．8 つの符号がすべて同一であれば境界面ではないため，ポリゴンは生成されない．生成されるポリゴンは 1 ボクセルあたり 1~4 枚である．ポリゴンの頂点は，ボ

図 4　出力結果

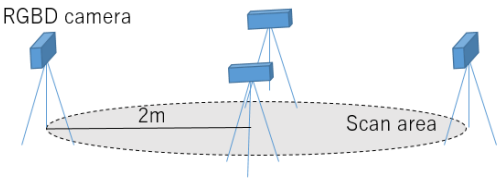

図 5　RGBD カメラの配置

クセルの頂点間の辺上に配置されるが，ボクセルデータの値により，三角形の頂点を変動させることで滑らかなモデルとなる．データを保持する全てのボクセルに対して表面形状推定を行うと全周の三次元モデルとなる．生成された三次元モデルを図 4 に示す．

ポリゴンの描画を行うために頂点情報や面法線ベクトルなどのポリゴン情報を配列で保持する必要がある．表面形状推定はボクセルごとに並列で行われており，ポリ

ゴン情報の格納も同時に行われる．しかし，ボクセルごとにポリゴン数が異なるため，格納先のアドレスが定まらない．そこで，各ボクセルが描画するポリゴン数の Prefix sum を求めることで並列にポリゴン情報の格納が可能となる．

5. 実装

本研究では RGBD カメラとして Kinect V2 を使用する．Kinect から得られるデータは 1920×1080 の RGB データと 512×424 の深度データであり，今回は表面形状推定に深度データを用いる． Kinect の台数を増やすと，より精緻なモデルが生成できると考えられるが，今回の実装環境では先行研究と同じ 4 台を使用する．図 5 のように，半径 2m の円周上に等間隔に Kinect を配置し，円の中心に被写体を配置する形をとった．

5.1. キャリブレーション

本研究では複数の RGBD カメラを用いる為，カメラ間のキャリブレーションが必要となる．リアルタイム性を重視している為，事前処理としてキャリブレーションを行う．

まず，適度に深度差のある剛体を各 RGBD カメラでスキャンし，深度マップを取得する．すべての深度マップが矛盾なく元の剛体を再現できるように重ね合わせることができれば，個々のカメラの位置を推定できたと言える．実際には複数の 3 つ以上の深度マップを同時に調整することは困難であるため，隣接する 2 つの深度マップで順に位置合わせを行う．位置合わせには ICP を用いる．ICP は対応関係が未知な 2 点群 $\boldsymbol{x}$と $\boldsymbol{y}$をマッチングさせ，反復的に比較していくことにより位置を合わせる手法である．任意の点 x_iと最近傍な y_iを探索し，それを仮の対応点とする．x_i と y_iの誤差 E が最小となるような$\boldsymbol{R}$と$\boldsymbol{t}$を求める．$\boldsymbol{R}$ は回転行列，$\boldsymbol{t}$は並進ベクトルである．誤差 E が閾値以下になるまで反復することで，変換行列を求めることができる．

$$E(\boldsymbol{R},\boldsymbol{t}) = \sum_{i=1}^{N} \|y_i - \boldsymbol{R}x_i + \boldsymbol{t}\|^2$$

しかし，一周分の位置合わせを終えた時に，最初と最後の深度マップが正しく重ね合わせることができない．これは， 1 つずつの位置合わせの誤差が積算されるためである．そこで，2 台間の位置合わせの後，全てのカメラ位置を微調整する．Kinect を N 台用いるとき，隣接するカメラ座標への変換行列をP_nとすると，$P_{1,\cdots,}P_N$の変換行列を掛け合わせると理論上は単位行列になるが，誤差の積算により差異が生じる．積算誤差行列はすべての変換行列の積の逆行列で表すことができる．その誤差行列の回転移動の角度を求め，N 分割し，全ての変換行列に掛けることで積算誤差の補間を行う．平行移動部分は要素を N 分割する．誤差行列の移動方向は基準となったカメラ座標系となるため，全ての変換行列に同じ誤差行列を用いることはできない．そこで，それぞれの Kinect を起点として周回の積算誤差を算出し，それらの値を用いてそれぞれの補間行列を求める．この補間行列を変換行列に乗算することで積算誤差を補う．

任意の RGBD カメラ座標をグローバル座標とすると，求めた回転行列と併進ベクトルにより，各カメラ座標系からグローバル座標系への変換が可能となる．グローバル座標系への変換は各カメラと接続しているクライアント PC で行うことにより，サーバ PC の処理を減らすことができる．

サーフェスモデルでは，RGBD カメラごとにモデルを生成し重ね合わせているため，僅かな位置合わせの誤差がそのまま三次元モデルに反映されてしまう．そのためフレーム毎に詳細な位置合わせ処理を行う必要があった．しかし，ボクセルベースでは，深度データは重みを与える処理にのみ用いるため，僅かなずれが生じても生成されるモデルへの影響はほとんどない．よってフレームごとの詳細な位置合わせ処理は不要となる．

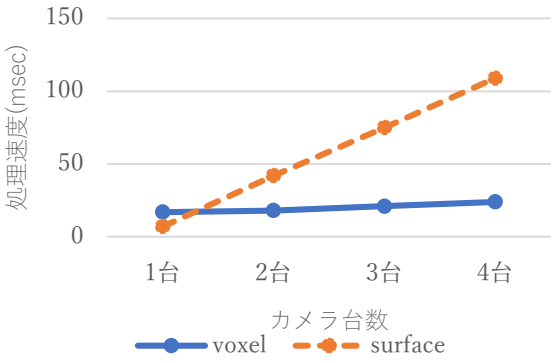

図 6　カメラ台数による処理速度の比較

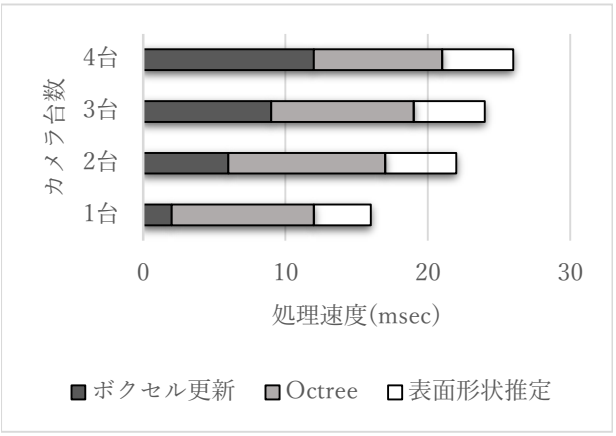

図 7　各処理の実行速度

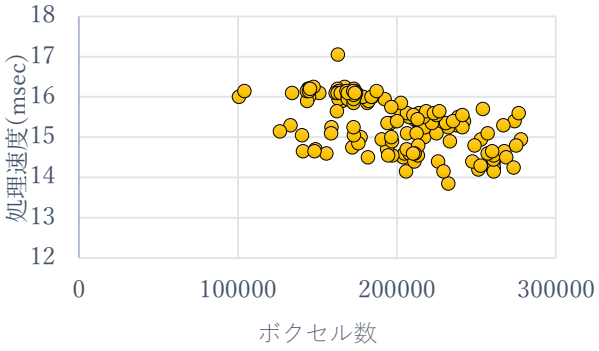

図 8　表面形状推定のボクセル数による処理速度

5.2. データ通信

本研究で用いる Kinect V2 では単一の PC で複数台を扱うことができない．そこで，各カメラから取得した深度データをサーバに送信することで同時に複数の RGBD カメラを扱えるようにする．また，サーバへの接続にはイーサネット，プロトコルはデータの欠落を避けるため TCP を採用する．

Kinect V2 では 0.8~8.0m の深度を取得することができるが，スキャンするのは円の中心に配置される被写体のみである．背景の物体など，円周外の深度データについては，送信前に除去することで通信コストの削減を行う．また，通信処理と以降の三次元モデル生成処理は並列に行われる．

6. 実験

提案手法の評価のために，下記の 3 種類の処理速度の測定を行う．

- 実験 1：RGBD カメラの台数による処理速度
- 実験 2：各処理の実行速度
- 実験 3：表面形状推定を行うボクセル数による処理速度

実験 1 では，使用する RGBD カメラを 1~4 台の場合の処理速度を測定し，サーフェスモデルベースと比較することでボクセルモデルベースの有意差を示す．実験 2 は 4.1 節のボクセルデータ更新，4.2 節の Octree による不要ボクセルの除去，4.3 節の表面形状推定のそれぞれの処理速度を測定する．また，これらの処理の実行速度がカメラ台数により変動するかを観測するために，1~4 台のカメラを用いて実験する．実験 3 では，被写体の大きさが変われば表面形状推定を行うボクセル数も変わると考え，被写体の大きさが描画速度に影響するかを検証する．実験方法は，一定時間スキャンを行っている間に動き回り，その時のポリゴン数と処理速度を記録する．

使用する RGBD カメラには Kinect V2 を用いた．RGBD カメラと接続し，データ送信を行うクライアントには Intel Corei7-4710MQ 2.50GHz を搭載した PC を，受信したデータから三次元モデルを生成するサーバには Intel Corei7-7700K 4.20GHz，NVIDIA GeForce GTX 1080Ti を搭載した PC を用いた．生成した三次元モデルの描画には OpenGL を用いる．実験環境は 4.2 節と同様とする．また，実験 1 と実験 2 については，被写体は描画面積の違いが処理速度に影響しないように，静止物体を用いる．

図 6 に実験 1 の比較結果を示す．破線は 3 節で行ったサーフェスベースの処理速度である．1 台では 17msec でサーフェスベースの 7msec のほうが早い結果となっているが，これは EM-ICP を行っていないためである．2 台以降はサーフェスベースでは詳細な位置合わせに EM-ICP を行っているためボリュームベースのほうが速くなっている．2 台で 18msec，3 台で 21msec，4 台で 24msec という結果になった．カメラが 1 台増加するごとに処理速度は 1~3msec 増している．

図 7 には実験 2 の結果を示す．ボクセルデータ更新処理は，1 台で 2msec，2 台で 6msec，3 台で 9msec，4 台で 12msec となった．各 RGBD カメラから取得した深度データを用いて更新処理を行っているため，カメラの台数が増加すると処理速度も増加する．しかし，Octree による不要空間の除去が 9~11msec，表面形状推定が 4~5msec となり，カメラの台数に関わらず処理速度はほぼ一定となっている．

図 8 に実験 3 の結果を示す．ボクセル数は約 100000~300000 であったが，いずれも 14~17msec という結果になった．

7. 考察

実験 1 より，提案手法のボクセルベースによる三次元モデル生成では，RGBD カメラの台数が 4 台でも 1 フレームの処理時間が 26msec となった．これはサーフェスベースでの処理時間の 109msec と比べ，4 分の 1 の速度となっている．また，サーフェスモデルはカメラの台数が増えるごとに速度の低下がみられたが，提案手法はほぼ一定の速度での処理が可能である．サーフェスモデルではカメラ間の詳細な位置合わせを 1 フレームごとに行っており，この処理が約 30msec ほどかかっており，また，位置合わせはカメラ間で行うため，カメラの台数が増えるごとに処理速度も低下してしまう問題があった．これに対して提案手法のボクセルモデルは，カメラごとに行う処理が 3.1 節で述べたボクセルデータの更新のみであり，その処理速度も 1 台あたり約 3msec となっている．そのため，カメラ台数が増加しても処理速度は大きく変わることがなかった．

また，実験 2 では Octree による不要ボクセルの除去と，表面形状推定処理はどちらもカメラの台数に関わらずほぼ一定の処理時間が測定された．この 2 つの処理は今後カメラの台数を増やしてもほぼ同じ速度での処理が可能と考えられる．ボクセルデータの更新処理はカメラの台数により処理速度が変動したが，1 台増加につき低下した速度が 3msec となった．しかし，30fps を実現するための処理速度は 33msec/frame 以下である必要があり，この数値と比較すると小さい値とは言えない．しかし，4 台使用時でも 30fps以上の描画が可能な処理速度であり，人物のリアルタイムモデル生成には実用可能である．また，更にカメラ台数を増やした時，例えばカメラを 8 台使用しても処理速度は 39msec 程度になると推測できる．フレームレートに換算すると約 24fps であり，アニメ映像のフレームレートと同値である．したがって，カメラ台数が 8 台以下であれば，日常で観ている映像作品と同等のフレームレートでの三次元描画が可能である．

実験 3 からは表面形状推定を行うボクセルの数に関わらず，ほぼ一定の速度で処理を行えることがわかる．被写体である人間が動き回ったり，撮影空間内にオブジェクトを持ち込んだりと描画面積が変動することが考えられるが，実験結果よりフレームレートにばらつきがでることはないと言える．

本研究では用途をテレイマージョンとして人物のモデル生成を前提としていたが，ボリュームボクセルのサイズを変えることで別の用途への応用が可能と考えられる．例えば，撮影空間を広げれば，部屋全体のモデルを生成することができ，仮想空間での部屋の再現や，防犯システムなどに活用できる．また，人物のようなサイズのモデル生成でも，ボリュームボクセルの分割数を細かくすれば，よりリアルで滑らかな三次元モデルの生成も可能となる．本研究では深度データの精度の問題で細かくすることができなかったが，カメラの台数を増やすことで被写体までの距離を縮めるなどすればモデルの欠落の心配もなくなる．

また，本研究では色情報を扱うことができなかったが，生成した三次元モデルにテクスチャマップを貼ることができればよりリアルな三次元モデルとなるため，今後の課題である．

8. まとめ

本研究では複数の RGBD カメラを用いた，ボクセルモデルベースのリアルタイム三次元モデル生成手法を提案した．各 RGBD カメラから得られる深度データをボクセルに対する重みとして用いることにより，カメラの台数が増加しても処理速度が大きく低下しない結果となった．カメラを 4 台使用時には 26msec/frame で三次元モデルをリアルタイムに生成することが可能で，既存のサーフェスベースのモデル生成手法と処理速度を比較すると約 4 分の 1 の速度となった．

文　献

[1] Newcombe, Richard A., et al. "KinectFusion: Real-time dense surface mapping and tracking." Mixed and augmented reality (ISMAR), 2011 10th IEEE international symposium on. IEEE, 2011.

[2] Izadi, Shahram, et al. "KinectFusion: real-time 3D reconstruction and interaction using a moving depth camera." Proceedings of the 24th annual ACM symposium on User interface software and technology. ACM, 2011.

[3] Alexiadis, Dimitrios S., Dimitrios Zarpalas, and Petros Daras. "Real-time, full 3-D reconstruction of moving foreground objects from multiple consumer depth cameras." IEEE Transactions on Multimedia 15.2 (2013): 339-358.

[4] Besl, Paul J., and Neil D. McKay. "A method for registration of 3-D shapes." IEEE Transactions on pattern analysis and machine intelligence 14.2 (1992): 239-256.

[5] Tamaki, Toru, et al. "CUDA-based implementations of Softassign and EM-ICP." Proc. of the 23rd IEEE Conference on Computer Vision and Pattern Recognition (CVPR2010) CD-ROM. 2009.

[6] Lorensen, William E., and Harvey E. Cline. "Marching cubes: A high resolution 3D surface construction algorithm." ACM siggraph computer graphics. Vol. 21. No. 4. ACM, 1987.

[7] 松本依里紗, and 藤田悟. "D-12-69 3 次元動体のリアルタイムスキャニング (D-12. パターン認識・メディア理解 B (コンピュータビジョンとコンピュータグラフィックス), 一般セッション)." 電子情報通信学会総合大会講演論文集 2016.2 (2016): 138.

[8] 松本依里紗, and 藤田悟. " 複数の深度カメラを用いたリアルタイム 3D スキャニング." 第 79 回全国大会講演論文集 2017.1 (2017): 105-106

[9] 松本依里紗, 藤田悟, and 廣津登志夫. " 複数の RGB-D カメラによるリアルタイムモデリング." 情報処理学会研究報告コンピュータビジョンとイメージメディア (CVIM) 2017.11 (2017-CVIM-209) (2017): 1-5.