マルチカメラを用いた三次元形状計測手法の並列計算による高速化

平成19年度 修士学位論文

マルチカメラを用いた三次元形状計測手法の並列計算による高速化

東北大学大学院 情報科学研究科
情報基礎科学専攻
博士課程前期2年の課程
堀口 研究室

A6IM1028
高橋一樹

平成20年 2月 27日

目 次

図 目 次

表 目 次

第1章　序論

第 1 章では，本研究の背景と目的を述べ，終わりに本論文の構成を述べる．

1.1　本研究の背景

三次元形状計測は，物体認識，仮想視点画像の生成など多くの分野で必要とされている技術である．その中でも，カメラによって撮影した画像を用いて三次元形状計測を行う手法があり，代表的なものとしてステレオ法がある．ステレオ法は，撮影した 2 枚の画像で一方の画像中の点が他方の画像中の点のどこに対応するか対応点を決定し，三角測量の原理で対象の三次元情報を得る手法である．カメラで撮影した画像のみを計測に用いるため，計測対象・計測環境への制限が少ないという特徴を持ち，種々の応用へ向けた様々な研究がなされている．しかし，ステレオ法には,「対応点決定の曖昧さ」と「遮蔽による誤対応」という二つの大きな問題があり，より実用的なステレオ法を実現するためには，これらの問題の解決を図る必要がある [1, 2].

「対応点決定の曖昧さ」は，各画像中の小領域のパターンや特徴点の一致度が最も高い部分を探索し対応点と決定するために生じる問題である．撮影した画像によっては対応点以外にも似たパターンや特徴をもった箇所が様々な所に存在する可能性は高く，また，同じ対象物を撮影していても，その左右の画像は視点の違う投影による幾何学的な歪み，カメラの特性の違い，ノイズ等により，完全に一致することはない．このように，画像間の対応点決定には曖昧さが存在し，対象が限定されていない一般のステレオ画像で，画像上の全ての部分の対応点を正確に求めることは困難となる．

「遮蔽による誤対応」は，画像を撮影したときの視点の違いにより見えている部分が各画像で異なるため，一方の画像で写っている部分が，他方の画像では遮蔽されて見えていないために，画像同士で対応点が決定できないことにより生じる問題である．遮蔽が生じている箇所は，対応点を決定する際に誤対応が発生し，正確な三次元形状計測が困難となる．このような遮蔽は，物体境界では必ず起こる現象である．このため，三次元形状計測の結果を仮想視点画像の生成に利用する場合など，対象物体の境界を正確に求める必要がある応用分野では，どのように遮蔽へ対処するかが重要な課題となっている．

「対応点決定の曖昧さ」に対しては，マルチカメラを利用した手法の有効性が

広く確認されている．また,「遮蔽による誤対応」は，三次元空間中の物体を扱うことで生じる現象であり，本質的に遮蔽の問題を解決するためには三次元情報を利用することが必要である．そのため,「対応点決定の曖昧さ」と「遮蔽による誤対応」の問題に対して，マルチカメラを用いた三次元形状計測手法に対し，三次元情報を効果的に導入することで両方の問題の解決を図る手法が提案されている．代表的な手法に，スペースカービング法 [3] や三次元情報の整合性を考慮した手法 [4] がある．

スペースカービング法は，三次元空間をボクセルに分割し，各ボクセルが物体境界に対応するか否かを順次判定することで三次元形状を計測する手法である．また，三次元情報の整合性を考慮した手法は，三次元空間をボクセルに分割し，各ボクセルが可視となる (遮蔽されない) 確からしさを計算して画素の一致度の判定に利用し，ボクセルが物体境界に対応するか否かを他のボクセルの状態を参照しながら決定していく手法である．これらの手法により，精度のよい形状計測を行うことができるが，マルチカメラ，三次元情報を利用するために扱うデータ量が増え，計算時間が増加してしまうという問題がある．特に，三次元情報の整合性を考慮した手法では，各ボクセル全てが他のボクセルの状態を参照して，反復更新を繰り返すため計算時間が膨大になってしまう．そのため，形状計測速度を向上させるための技術が必要になる．スペースカービング法については，各ボクセルに対する判定処理を並列化し，GPU(Graphics Processing Unit) の機能を用いて高速に処理する手法が提案されている [5, 6]．しかし，三次元情報の整合性を考慮した手法のように，他のボクセルの状態を参照しながら物体境界に対応するか否かを決定していく手法も多く [4, 7, 8, 9]，他のボクセルの状態の参照に要する通信量を低減する並列化手法が必要になる．

1.2 本研究の目的

本研究では，マルチカメラを用いたボクセルベースの三次元形状計測手法に対し，ボクセル空間を視点とボクセルを通る直線の傾きを用いて分割を行うことで通信量を抑えた新たな並列化手法を提案する．具体的には，複数のカメラの視点から，各視点との距離が最小になる直線と視点の平均を求め，その直線を軸として視点の平均と各ボクセルを通る直線の傾きを求め，その傾きの大きさの順にソートしてボクセル空間を分割する．これにより，並列化の際に生じる通信量の増加という問題に対処し，三次元形状計測手法の高速化を図る．さらに，この並列化手法を改善するための通信方法，計算負荷についても考察し，さらなる高速化を図る．

1.3　本論文の構成

本論文では，以下の構成に従って論ずる．第 2 章では，ボクセルベースの三次元形状計測手法について述べ，その概要と問題点を挙げる．第 3 章では，ボクセルベースの三次元形状計測手法について，第 2 章で挙げた問題点を解決するための新たな並列化手法を提案し，さらに，実装を行いその有効性を示すための実験および考察を述べる．第 4 章では，提案した並列化手法を，より改善するための通信方法，計算負荷について考察し，その有効性を示すための実験および考察を述べる．最後に，第 5 章において結論を述べる．

第2章　ステレオ法に基づく三次元形状計測手法

2.1　はじめに

第 2 章では，ボクセルベースの三次元形状計測手法について述べる．まず，三次元形状計測の基本となるステレオ法の原理について説明し，その問題点を述べる．次に,「遮蔽による誤対応」の問題に対処する手法である三次元情報を利用した協調ステレオアルゴリズムの概要を述べる．さらに,「対応点決定の曖昧さ」と「遮蔽による誤対応」の両方の問題に対処する手法である三次元情報とマルチカメラを利用したスペースカービング法と三次元情報の整合性を考慮した手法の概要を述べる．最後に，ボクセルベースの三次元形状計測手法の問題点を指摘し，本章のまとめを述べる．

2.2　ステレオ法

ステレオ法は，同一の対象を撮影した 2 枚の画像で対応点を決定し，三角測量の原理でカメラから対象までの奥行きを求める手法である．本節では，ステレオ法の計測原理と対応点探索の手法について説明し，その問題点を挙げる．

2.2.1　計測原理

ステレオ法は，異なる視点から同一の対象を撮影し，それぞれの画像上への投影位置の違い (視差) を利用して，三角測量の原理に基づき三次元情報を得るものである．ステレオ法の最も単純な構成は，図 2.1 のように，2 台の焦点距離の等しいカメラが平行で，撮影される画像面が同一平面上に乗るように配置された場合である．図 2.1 のように，三次元空間の座標系 (X, Y, Z) と，それぞれの画像における画像座標系 (x, y) を定めたときに，点 $P(X, Y, Z)$ がそれぞれの画像上の点 $p_1(x_1, y_1)$, $p_2(x_2, y_2)$ に投影されたとすると，次の式が成立する．

$$X = \frac{B(x_1 + x_2)}{2d} \tag{2.1}$$

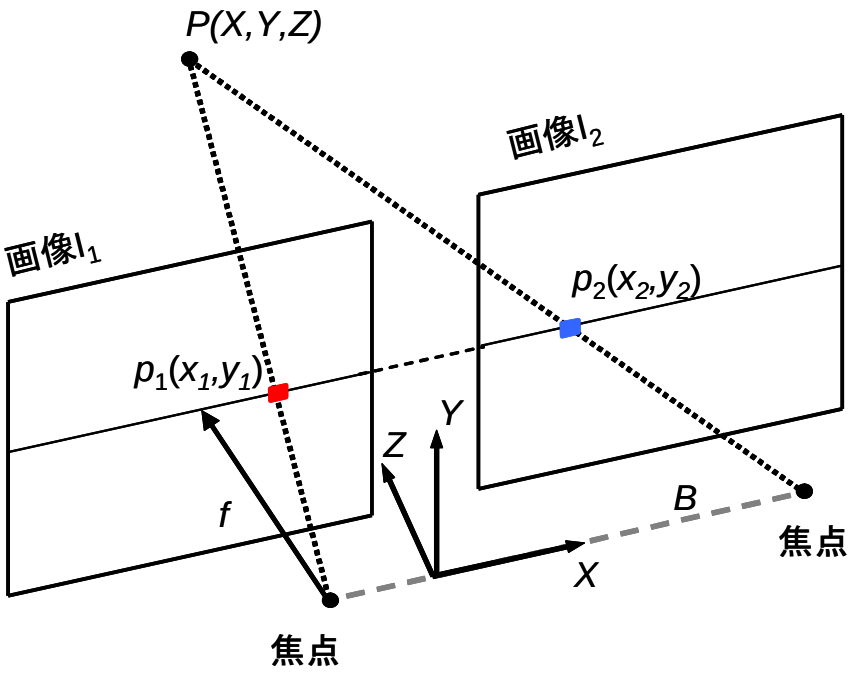


図 2.1: ステレオ法の基本的な配置

$$Y = \frac{B(y_1 + y_2)}{2d} \tag{2.2}$$

$$Z = \frac{Bf}{d} \tag{2.3}$$

$$d = x_1 - x_2 \tag{2.4}$$

ここで，f はカメラの焦点距離，B はカメラの焦点間距離であり，d は視差である．従って f，B，が既知であれば，左右の画像から三次元位置 $P(X, Y, Z)$ を計算することができる．カメラが平行でない場合や，カメラのレンズによる画像の歪みが生じる場合には，ステレオ画像をあたかも平行なカメラで撮影したかのように画像を変換する平行化や，レンズによる画像の歪み補正を行うことで，カメラが平行なステレオ画像と同じように考えることが可能である．

2.2.2 対応点探索

前小節で述べたように，三次元空間中の点 P が，ステレオ画像に投影された点を決定できれば，三次元位置を計測することが可能となる．これは，一方の画像中の点 p_1 が，他方の画像中のどの点に対応するかを決定できれば，p_1 に対応する

三次元空間中の点の三次元位置を計測できることと同義である．よって，ステレオ法による三次元形状計測では，一方の画像中の点が他方の画像中のどの点に対応するかを決定することが重要になる．

探索する範囲

図 2.1 の配置では，対応点である $p_1(x_1, y_1)$，$p_2(x_2, y_2)$ は同じ垂直座標をもち，

$$y_1 = y_2 \tag{2.5}$$

であり，水平座標のみが異なる．そのため，対応点を探索する際には，同一垂直座標上を探索すればよい．一般的にも，一方の画像中の点に対応する点は，他方の画像中のある直線上のどこかに必ず存在する．このような直線をエピポーラ線という．

対応点決定

図 2.2 のように，対象を 2 つの異なる視点から平行なカメラで撮影した場合の対応点を決定する方法について述べる．以下では，対応点決定の一手法として，各画像中の小領域で輝度値の差を比較することにより対応点の決定を行う手法について説明する．

基準となる画像を I_1 とし I_1 内の注目点 $I_1(x_1, y_1)$ に対して，その画素を中心とした矩形領域 (ウィンドウ)R を設定し，比較する画像 I_2 内のエピポーラ線上の各画素で輝度値の差の 2 乗和 (SSD:Sum of Squared Difference) を計算する．画像 I_1 内の画素 $I_1(x_1, y_1)$ に対する画像 I_2 内の探索範囲を a とし，I_2 の対応画素を $I_2(x_2(a), y_2(a))$ とすると，$I_1(x_1, y_1)$ に対する SSD は，

$$\mathrm{SSD}(x_1, y_1, a) = \sum_{(u,v)\in R} (I_1(x_1 + u, y_1 + v) - I_2(x_2(a) + u, y_2(a) + v))^2 \tag{2.6}$$

で求められる．SSD はウィンドウ内の輝度値が近いほど小さくなり，完全に一致すれば 0 となる．このように，基準画像 I_1 内の画素 $I_1(x_1, y_1)$ に対して，比較画像 I_2 で同じ箇所が映っている画素の輝度値が近くなるため，SSD が最小となる点を対応点と決定する．

ステレオ法の問題点

ステレオ法には,「対応点決定の曖昧さ」と「遮蔽による誤対応」という 2 つの大きな問題点がある．

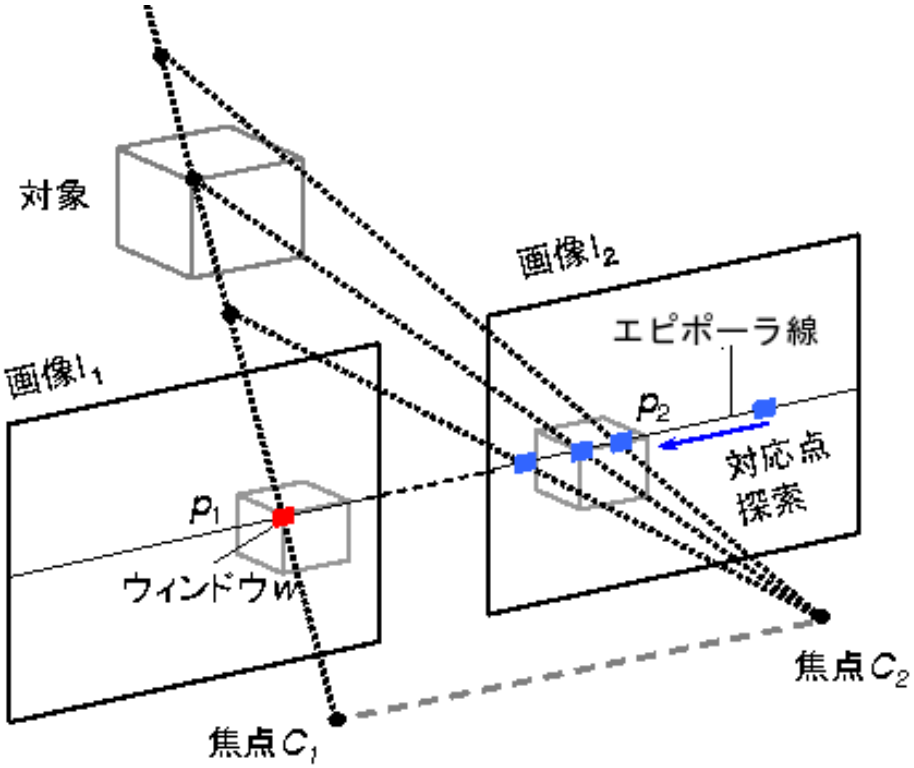


図 2.2: 対応点探索

- 対応点決定の曖昧さ

  ステレオ法は SSD が最も小さくなる箇所を探索するが，一般的には，撮影したステレオ画像によって対応点以外にも類似した箇所が複数存在する可能性は高い．そのため，対応点を一意に決定できずに誤対応が生じることがある．図 2.3 のように，画像 I_1 の注目点 p の探索の際に，画像 I_2 で真の対応点 p_2 以外にも点 p と輝度値が近い点 p_1，p_3 があると，SSD が p_2 で必ずしも最小とはならず，誤対応が生じ，正確な三次元形状計測が困難になる．この問題を解決する手法には，3 台以上のカメラで対象を撮影し，対象に関する冗長な情報を用いて対応点探索を行うマルチカメラを利用したステレオ法や，探索を行う点の周囲の情報を有効に利用して対応点探索を行う手法などが提案されている． [10, 11, 12]

- 遮蔽による誤対応

  ステレオ法は，画像の撮影された視点の違いを利用して三次元形状計測を行うが，それにより遮蔽の問題が生じる．遮蔽が生じている箇所は，図 2.4 のように，注目点 p が一方の画像 I_1 では見える領域となり，他方の画像 I_2 では見えない領域となる．このような p が存在する領域は，真の対応点を探索することが不可能であるにも関わらず，SSD が最小の点を対応点として決定

してしまうため誤対応が発生する．このような遮蔽の問題に対処する手法については多くの研究がなされており [1, 2, 13, 14, 15, 16]，三次元情報を利用することで遮蔽への対応を図る手法や，撮影された画像から得られる二次元情報を用いて遮蔽を検出したり，対象の三次元形状に関する仮定やモデルを用いる手法が提案されている．

2.3 ボクセルベースの三次元形状計測手法

前節で説明したステレオ法の問題点である「対応点決定の曖昧さ」,「遮蔽による誤対応」への対応方法の一つとしてボクセルベースの三次元形状計測手法がある．ボクセルベースの三次元形状計測手法は，計測対象の周りの三次元空間をボクセルに分割し，異なる 2 つの (または複数の) カメラから撮影された画像を基に，各ボクセルが物体境界に対応するか否かを判定することで三次元形状計測を行う手法である．本節では，ボクセルベースの三次元形状計測手法である協調ステレオアルゴリズム，スペースカービング法，三次元情報の整合性を考慮した手法について述べる．

2.3.1 協調ステレオアルゴリズム

協調ステレオアルゴリズムは，図 2.5 に示すように，ボクセルに分割したステレオ視差空間を用い，計測の中間結果として得られた視差情報を処理に導入するステレオ法である．

協調ステレオアルゴリズムでは，まず，画像の高さ r，幅 c，ステレオ画像の視差 d からなる視差空間をボクセル $V\,(r, c, d)$ に分割し，各ボクセル $V(r, c, d)$ に，そのボクセルが物体境界に対応する確からしさ $L_t(r, c, d)$ を設定する．確からしさの初期値 $L_0(r, c, d)$ は次のように決定する．

1. 各ボクセル $V(r, c, d)$ に対応する画像上の点 $p_1(r, c)$ と $p_2(r, c)$ の画素値の一致度 (輝度値の差など) を計算

   (物体境界に対応するボクセル $V(r, c, d)$ では，対応する画像上の点 $p_1(r, c)$ と $p_2(r, c + d)$ が類似した画素値を持つことになり，画素値の差は小さくなる．)

2. 物体境界に対応する可能性が高いほど (画素値の差が小さいほど) 大きくなるような確からしさの初期値 $L_0(r, c, d)$ を計算 $(0 \leq L_0 \leq 1)$

次に，以下の式を用いて，ボクセル $V(r, c, d)$ の確からしさ $L_t(r, c, d)$ を伝播させることで，各ボクセル $V(r, c, d)$ の確からしさ $L_t(r, c, d)$ の更新を行う．

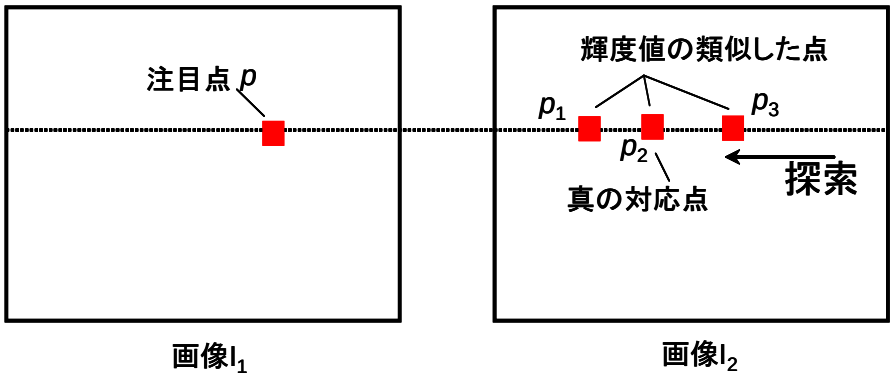


図 2.3: 対応点決定の曖昧さ

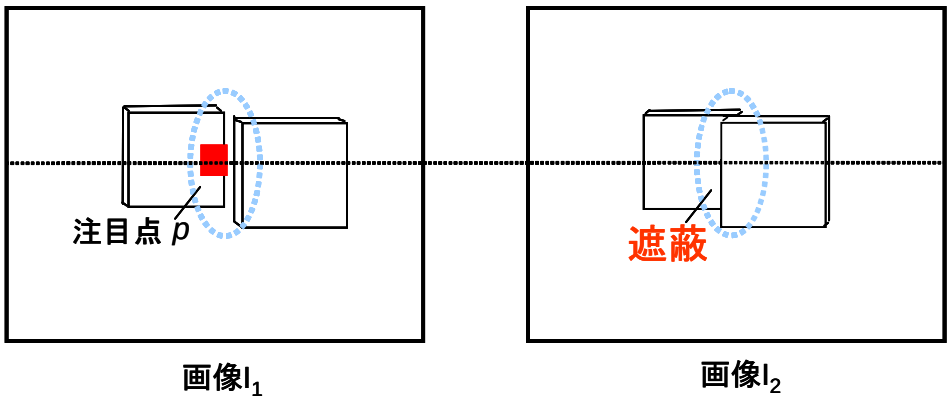


図 2.4: 遮蔽による誤対応

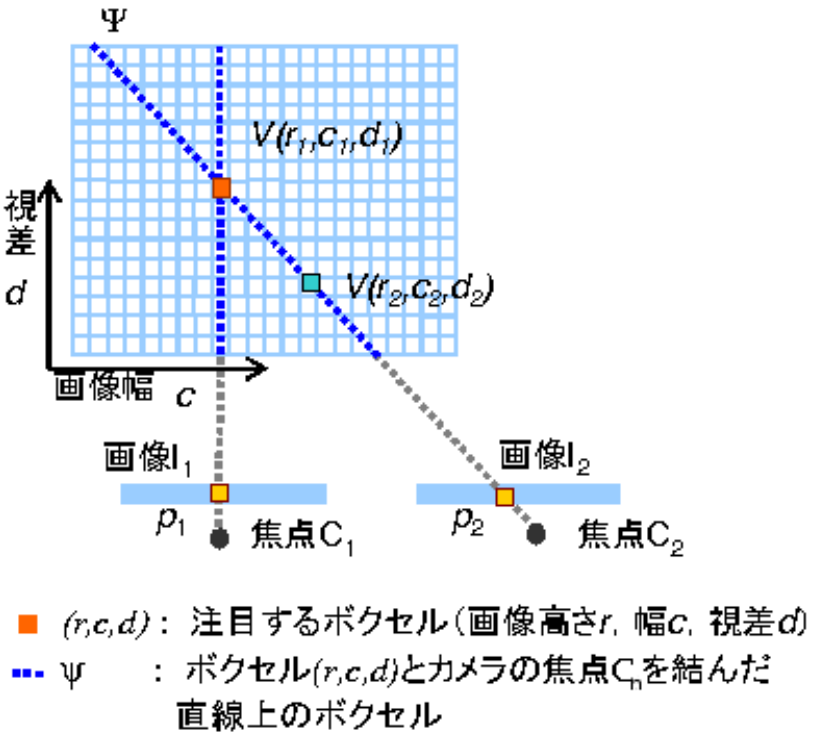


図 2.5: 協調ステレオアルゴリズム

$$L_{t+1}(r,c,d) = L_0(r,c,d) \times \left(\frac{L_t(r,c,d)}{\sum_{(r',c',d')\in\Psi} L_t(r',c',d')} \right)^{\alpha} \tag{2.7}$$

ここで，$\alpha > 1$，注目ボクセル $V(r,c,d)$ と各画像の焦点 C_n を通る直線上に存在するボクセルの集合を Ψ とする．注目ボクセル $V(r,c,d)$ の確からしさ $L_t(r,c,d)$ を Ψ 中のボクセルの確からしさの和 $\sum_{(r',c',d')\in\Psi} L_t(r',c',d')$ で割った値を掛けることで，確からしさ $L_t(r,c,d)$ の高いボクセルでは確からしさはそれほど小さくならず，確からしさ $L_t(r,c,d)$ の低いボクセルでは確からしさはより小さくなるように更新される．確からしさ L_t を更新することで，例えば，図 2.5 の $V(r_1,c_1,d_1)$ を注目ボクセルとしたときに，$V(r_1,c_1,d_1)$ と焦点 C_2 を通る直線上のボクセルである $V(r_2,c_2,d_2)$ の確からしさ $L_t(r_2,c_2,d_2)$ が高いならば，$V(r_1,c_1,d_1)$ の確からしさ $L_t(r_1,c_1,d_1)$ は小さくなる．このように，協調ステレオアルゴリズムでは遮蔽されていると考えられるボクセルでは確からしさを低下させることができる．また，各ボクセルにおける確からしさ L_t を他のボクセルに伝播されることで，視差空間全体で整合がとれるように L_t を修正することが可能となっている．しかし，協調ステレオアルゴリズムは，二枚の画像を用いたステレオ法にしか対応しておらず，「対応点決定の曖昧さ」が残り，遮蔽が生じている箇所では物体境界に対応するボ

クセルを探索できない．

2.3.2　スペースカービング法

スペースカービング法は，三次元空間をボクセルに分割し，各ボクセルが物体境界に対応するか否かを順次判定することで三次元形状を計測する手法である．

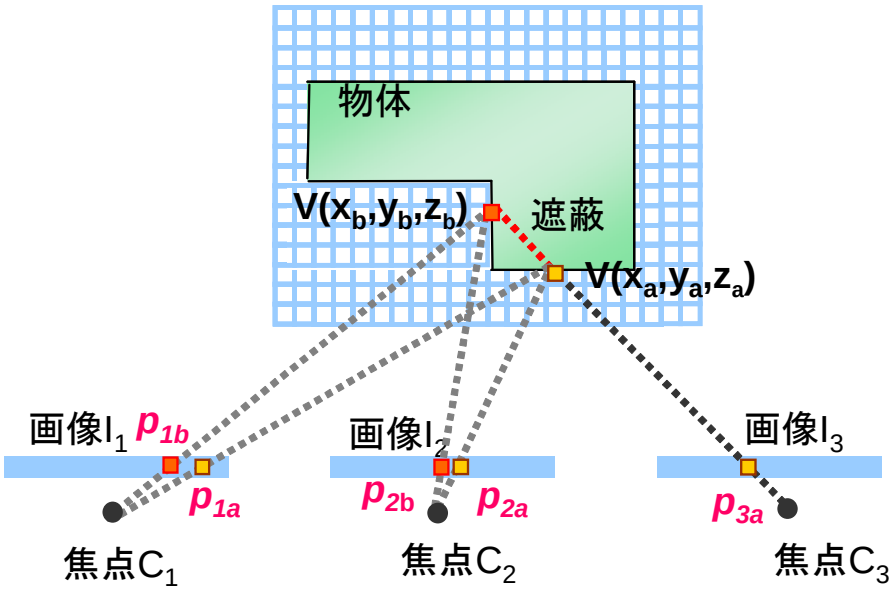


図 2.6: スペースカービング法

計測対象の三次元空間にボクセル $V(x, y, z)$ を定義し，ある $V(x, y, z)$ と各カメラの焦点 C_n を結ぶ直線を各々求め，各カメラで撮影された画像へ $V(x, y, z)$ が投影される位置を決定する．そして，全ボクセルについて投影された各画素間で画素の一致度（輝度値の差など）を求める．もし，$V(x, y, z)$ が物体境界に対応し，かつ，全てのカメラから見える場合，各画像上の対応する画素は同様な輝度値を持つはずであり，各画素間での一致度は高くなる．一方，$V(x, y, z)$ が物体境界に対応しない場合，あるいは，一部の画像で遮蔽が生じている場合には，各画素での一致度が低くなる．このように，各ボクセルで計算された一致度は，そのボクセルの対応する箇所がどのような状態にあるかを判断する尺度となる．そのため，ある閾値より一致度が大きい（輝度値の差が小さい）ボクセルを物体境界に対応するボクセルと判定し，全ボクセルに対して物体境界に対応するか否か順次判定を行うことで三次元形状計測を行う．

物体境界に対応すると判断されたボクセルの三次元情報を利用することで，遮蔽が生じている箇所が判断可能となる．これにより，遮蔽が生じている画像による影響を除いて一致度を計算し直すことで，各ボクセルが物体境界に対応するか

否かを判定する際に，遮蔽が与える影響を抑えることが可能になる．具体的には，ボクセル $V(x,y,z)$ とカメラの焦点 C_n を結ぶ直線上に，物体境界に対応すると判定された他のボクセルが存在する場合，C_n で撮影された画像中では $V(x,y,z)$ が遮蔽されることになる．そこで，$V(x,y,z)$ での一致度を求める際には，C_n で撮影された画像を使用しないことにする．例えば，図 2.6 において，点 $V(x_a,y_a,z_a)$ に位置するボクセルが物体境界に対応すると判断されたとする．点 $V(x_b,y_b,z_b)$ は，カメラの焦点 C_3 と $V(x_a,y_a,z_a)$ を通る直線上に位置するため，C_3 で撮影された画像 I_3 中では，$V(x_b,y_b,z_b)$ が $V(x_a,y_a,z_a)$ により遮蔽されることなる．そこで，$V(x_b,y_b,z_b)$ に対応するボクセルでの一致度を求める場合，I_3 を用いた一致度は使用せず，カメラ C_1,C_2 で撮影された画像 I_1,I_2 のみを用いた一致度を計算する．これにより，遮蔽の影響で一致度が小さくなることを防ぎ，$V(x_b,y_b,z_b)$ が物体境界に対応すると正しく判断されるようにする．このように，スペースカービング法では，遮蔽が生じている箇所に対する遮蔽されている画像による影響を除き，遮蔽されて物体境界と決定されなかったボクセルを物体境界と判定可能となるような処理を行っている．しかし，ボクセルを一度物体境界であると決定すると，その判断を変更できないため，協調ステレオアルゴリズムのようにボクセル全体での整合を考慮した処理を行っておらず，誤って物体境界でないボクセルを物体境界と判定してしまう可能性がある．また，物体境界であるボクセルの判定は閾値処理によって行われるために曖昧さが存在するという問題がある．

2.3.3 三次元情報の整合性を考慮した手法

スペースカービング法では，ボクセル全体での整合を考慮した処理を行っていないため，物体境界の判定誤りを生じる可能性がある．この問題を解決するための手法として，本小節では，マルチカメラを用いたボクセルベースの三次元形状計測手法である三次元情報の整合性を考慮した手法 [4] について述べる．この手法では，各ボクセル $V(x,y,z)$ について以下のような値

- $w_n(x,y,z)$：$V(x,y,z)$ が画像 I_n で可視となる (遮蔽されない) 確からしさ
- $S(x,y,z)$：$V(x,y,z)$ を各画像に投影した箇所の画素の一致度の総和
- $e(x,y,z)$：$V(x,y,z)$ が物体境界に対応する確からしさ

を更新する処理と，更新された値に基づき物体境界に対応するボクセルを決定する処理からなる．

処理の流れは以下の Step0 から Step3 からなる．

Step 0. 処理の開始 (物体境界に対応する確からしさを $e(x,y,z)=0$ とする)

Step 1. 各ボクセル $V(x,y,z)$ の $e(x,y,z)$ の設定を行うため，以下の Step 1.a 1.b 1.c の順に計算

a. $V(x,y,z)$ が画像 I_n で可視となる確からしさ $w_n(x,y,z)$ を計算

b. $V(x,y,z)$ を各画像に投影した箇所の画素の一致度の総和 $S(x,y,z)$ を計算

c. $V(x,y,z)$ が物体境界に対応する確からしさ $e(x,y,z)$ を計算

Step 2. ボクセル $V(x,y,z)$ の $e(x,y,z)$ を他のボクセルの状態を参照しながら反復更新

- 処理を継続する場合には Step1 へ戻る
- 処理を終了する場合には Step3 へ進む

Step 3. ボクセル $V(x,y,z)$ の $e(x,y,z)$ に基づき, 物体境界に対応するボクセルを決定

この処理の流れを図 2.7 に示す．Step1,Step2 による $e(x,y,z)$ の反復更新により，物体境界に対応する可能性の高いボクセルは $e(x,y,z)$ が上昇し，物体境界に対応する可能性の低いボクセルは $e(x,y,z)$ が低下する．反復により $e(x,y,z)$ は収束するため，最終的な $e(x,y,z)$ の値に基づき物体境界に対応するボクセルを決定する．

物体境界に対応する確からしさの計算

Step1,Step2 では，各ボクセル $V(x,y,z)$ の $e(x,y,z)$ を設定し，他のボクセルの状態を参照しながら $e(x,y,z)$ を反復更新する．以降では，$w_n(x,y,z), S(x,y,z), e(x,y,z)$ の計算について詳しく説明する．

なお，以下では，図 2.8，2.9 に示すように，各視点 o_n $(n = 1,2,3,\ldots,N)$ から撮影された画像を I_n とし，$V(x,y,z)$ を I_n へ投影した箇所を中心とする矩形領域を $R_n(x,y,z)$ で表す．また，$V(x,y,z)$ と各 o_n とを通る直線上に位置するボクセルの集合を $\phi_n(x,y,z)$，$V(x,y,z)$ と各 o_n とを結ぶ線分上に位置するボクセルの集合を $\phi'_n(x,y,z)$ とし，各 $V(x,y,z)$ に対する $\cup_{n=1}^{N}\phi_n(x,y,z)$ を $\Phi(x,y,z)$ で表す．

- 各ボクセル $V(x,y,z)$ が画像 I_n で可視となる確からしさ $w_n(x,y,z)$ を計算

  大きな $e(x,y,z)$ を持つボクセルが $\phi'_n(x,y,z)$ 内に存在する場合，$V(x,y,z)$ は，I_n において，そのボクセルにより遮蔽される可能性が高い．そこで，式 (2.8) により，可視となる確からしさ $w_n(x,y,z)$ を計算する．

$$w_n(x,y,z) = 1 - \frac{\max_{\phi'_n}(e(x,y,z)) - \mathrm{avg}_{\phi_n}(e(x,y,z))}{\max_{\phi_n}(e(x,y,z)) - \mathrm{avg}_{\phi_n}(e(x,y,z))} \tag{2.8}$$

  ここで，$\max_{\phi_n} e(x,y,z)$ と $\mathrm{avg}_{\phi_n} e(x,y,z)$ は，$\phi_n(x,y,z)$ 内での $e(x,y,z)$ の最大値と平均値を，$\max_{\phi'_n} e(x,y,z)$ は，$\phi'_n(x,y,z)$ 内での $e(x,y,z)$ の最大値を各々表す．

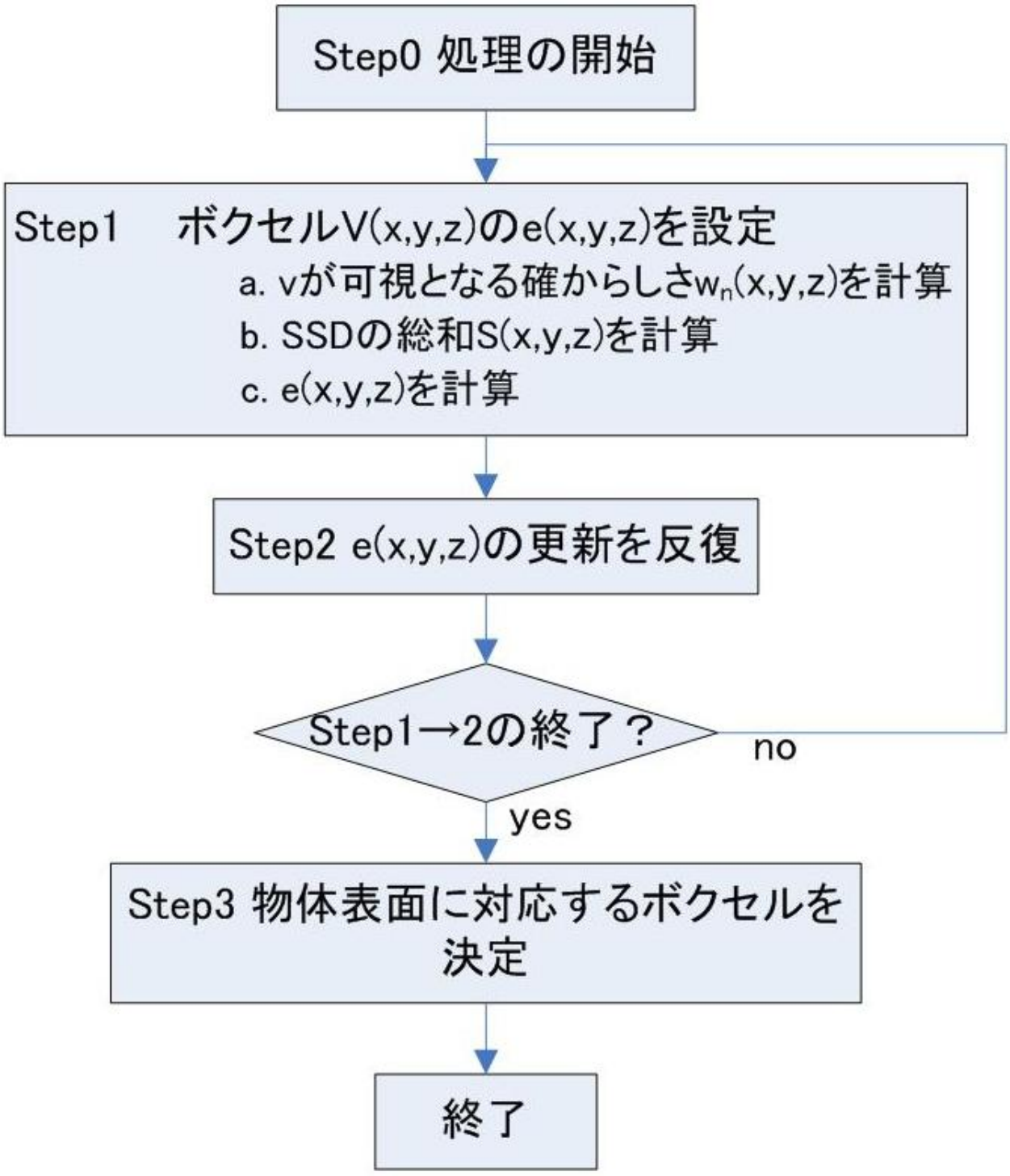

Step0 処理の開始
Step1　ボクセルV(x,y,z)のe(x,y,z)を設定
a. vが可視となる確からしさw_n(x,y,z)を計算
b. SSDの総和S(x,y,z)を計算
c. e(x,y,z)を計算
Step2 e(x,y,z)の更新を反復
Step1→2の終了？
no
yes
Step3 物体表面に対応するボクセルを決定
終了

図 2.7: フローチャート

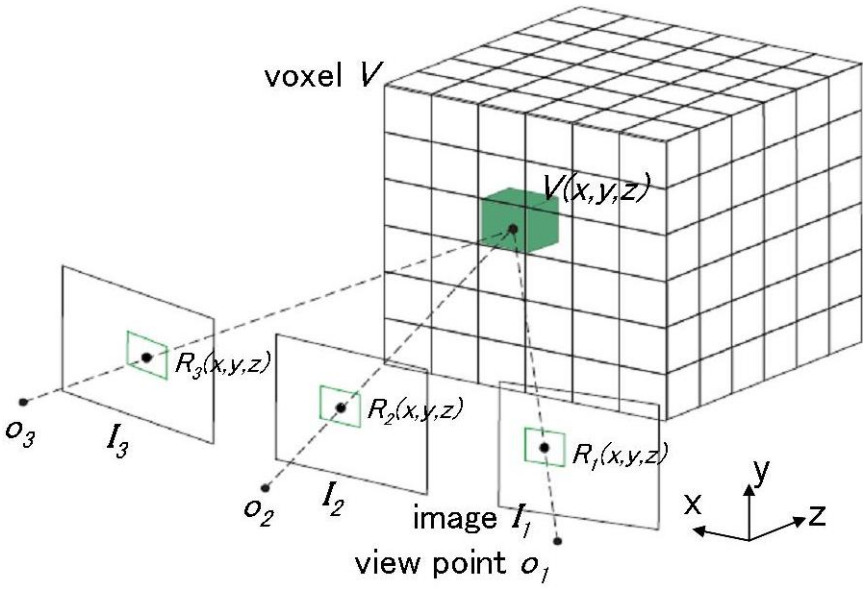


図 2.8: 視点とボクセルの関係

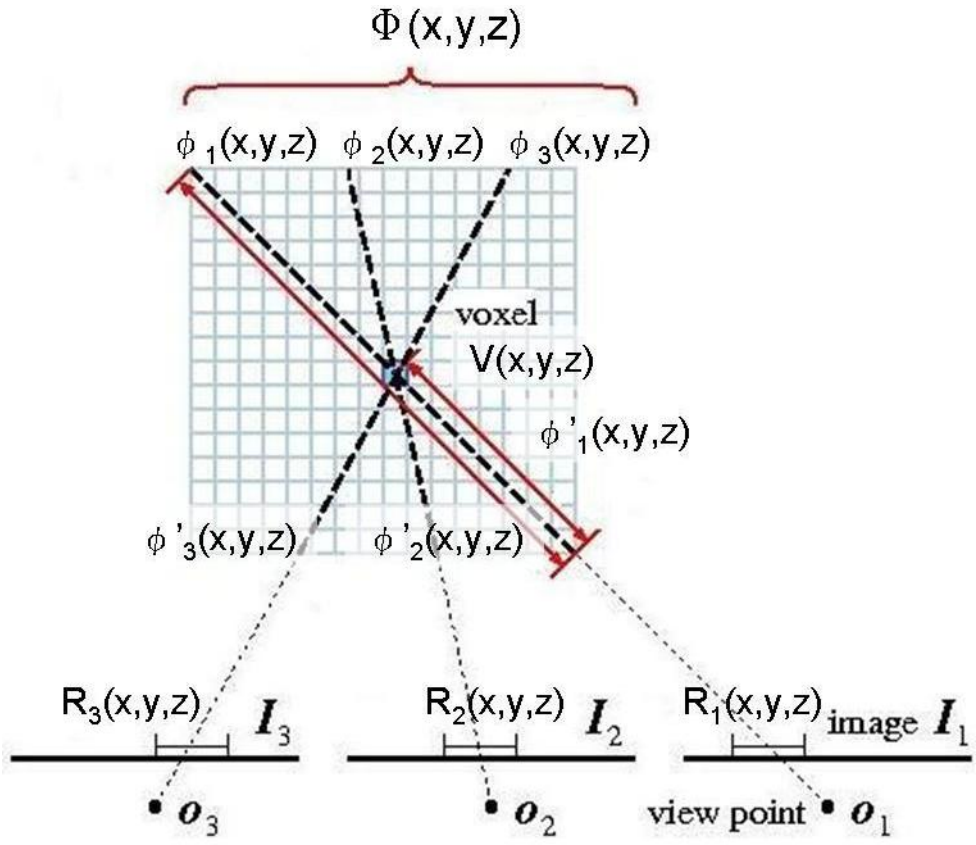


図 2.9: 視点とボクセルの関係（y 軸方向から見た図）

- 各ボクセル$V(x,y,z)$を各画像に投影した箇所の画素の一致度の総和$S(x,y,z)$を計算

  全ての画像対について，$V(x,y,z)$をI_nへ投影した箇所を中心とする矩形領域$R_m(x,y,z), R_n(x,y,z)$間の$\mathrm{SSD}_{m,n}(x,y,z)$を求め，式(2.9)により，画素の一致度$S(x,y,z)$を計算する．

$$S(x,y,z) = \frac{\sum\limits_{1\leq m<n\leq N} w_{m,n}(x,y,z) \times \mathrm{SSD}_{m,n}(x,y,z)}{\sum\limits_{1\leq m<n\leq N} w_{m,n}(x,y,z)} \tag{2.9}$$

  式(2.9)では，$V(x,y,z)$が可視となる確からしさが低い画像を用いたSSDへ小さな$w_{m,n}$を与えることで，遮蔽の影響により$S(x,y,z)$が増加することを防いでいる（$w_{m,n}$には，2つの重みw_m, w_nの内，値が小さいものを用いる）．

- 各ボクセル$V(x,y,z)$が物体境界に対応する確からしさ$e(x,y,z)$を計算

  $S(x,y,z)$が小さい$V(x,y,z)$は，各I_nで同じように見えており，物体境界に対応する可能性が高いと考えられる．そこで，式(2.10)により，物体境界に対応する確からしさ$e(x,y,z)$を求める．

$$e(x,y,z) = 1 - \frac{S(x,y,z) - \min_V S(x,y,z)}{\max_V S(x,y,z) - \min_V S(x,y,z)} \tag{2.10}$$

  ここで，$\max_V S$と$\min_V S$は，全ボクセル$V(x,y,z)$内での$S(x,y,z)$の最大値と最小値を各々表す．

- 物体境界に対応する確からしさ$e(x,y,z)$の反復更新

  Step2では，他のボクセルの状態を参照しながら物体境界に対応する確からしさ$e(x,y,z)$の反復更新を行う．注目する$V(x,y,z)$の$e(x,y,z)$が，$\Phi(x,y,z)$内で相対的に小さな値であるならば，その$V(x,y,z)$が物体境界に対応する可能性は低いと考えられる．そこで，$\Phi(x,y,z)$内での相対的な大きさに基づき，式(2.11)により，$e(x,y,z)$の値を更新する．

$$e_{t+1}(x,y,z) = e_0(x,y,z) \times \left(\frac{e_t(x,y,z) - \mathrm{avg}_\Phi e_t(x,y,z)}{\max_\Phi e_t(x,y,z) - \mathrm{avg}_\Phi e_t(x,y,z)} \right)^\alpha \tag{2.11}$$

  ここで，$e_0(x,y,z)$は，更新される初期値を表し，Step 1cで求めた$e(x,y,z)$の値を用いる．また，αは1より大きい定数である．$\max_\Phi e_t(x,y,z)$と$\mathrm{avg}_\Phi e_t(x,y,z)$は，$t$回目の更新時における$\Phi(x,y,z)$内での$e(x,y,z)$の最大値と平均値を各々表す．

物体境界に対応するボクセルの決定

注目する$V(x,y,z)$の$e(x,y,z)$が各$\phi_n(x,y,z)$上で極大となるかを調べ，極大となる$\phi_n(x,y,z)$における$w_n(x,y,z)$の総和が閾値γ以上であれば，$V(x,y,z)$が物体境界に対応するものと判定する．例えば，図2.9の例で，$e(x,y,z)$が$\phi_1(x,y,z), \phi_3(x,y,z)$上で極大となる場合，$w_1(x,y,z)+w_3(x,y,z)>\gamma$ならば，$V(x,y,z)$が物体境界に対応すると判断する．これにより，物体境界に対応する可能性が高く，かつ，複数の画像で可視となっている可能性も高いボクセルを物体境界として選択する．

このように，三次元情報の整合性を考慮した手法では，マルチカメラを用いることで「対応点決定の曖昧さ」の問題に対処し，また，$V(x,y,z)$が画像I_nで可視となる確からしさ$w_n(x,y,z)$を用いて，遮蔽が生じている箇所に対する遮蔽されている画像からのSSDの影響を少なくすることで,「遮蔽による誤対応」の問題に対処している．さらに，他のボクセルの参照と自身の状態の更新を反復することで，ボクセル全体の整合性もとっているため，非常に精度の高い三次元形状計測が可能となる．

2.4 ボクセルベースの三次元形状計測手法の問題点

前節で述べた例のように，カメラ画像を用いた三次元形状計測に三次元情報を保持したボクセルを用いることで,「遮蔽による誤対応」の問題に対処することができる．しかし，より良い精度で三次元形状計測をするためには，マルチカメラを用いたり，単純に，ボクセルサイズを密にするなどする必要があるが，いずれも扱う情報量が増加するため，計算時間が増加してしまう．さらに，他のボクセルを参照することで判定精度を向上させる手法では，各$V(x,y,z)$において，他のボクセルの状態の参照と自身の状態の更新を反復するため，さらに膨大な計算時間が必要となる．また，他のボクセルを参照する場合，注目するボクセル$V(x,y,z)$と各視点o_nを通る直線(または線分)上のボクセルを参照する必要があり，近傍のボクセルのみを参照するわけではないため，ボクセルの参照関係はかなり複雑になる．

2.5 まとめ

本章では，ステレオ法の「遮蔽による誤対応」と「対応点決定の曖昧さ」の問題に対処するためのボクセルベースの三次元形状計測手法について述べた.「遮蔽による誤対応」の問題に対処する手法である，協調ステレオアルゴリズムと,「遮蔽による誤対応」と「対応点決定の曖昧さ」の両方の問題に対処する手法である，スペースカービング法と三次元情報の整合性を考慮した手法の概要を述べた．そして，ボクセルベースの三次元形状計測手法の問題点を指摘した.

次章では，この問題点に対処するための新たな並列化手法を提案し，その有効性を示すための評価を行う．

第3章　三次元形状計測のボクセル領域の分割による並列化手法と計算時間の評価

3.1　はじめに

第3章では，ボクセルベースの三次元形状計測手法を並列計算により高速化する手法を提案し，その有効性を示すための実験を行い，実験結果を示し考察を行う．

3.2　並列化手法

3.2.1　並列化の際の必要な通信について

第2.3.3小節で説明した三次元情報の整合性を考慮した手法では，$V(x,y,z) \in V$ である一つ一つのボクセル $V(x,y,z)$ に対して各ステップの計算が行われるため，ボクセル空間を $P(P \geq 1)$ 分割して P 個のボクセル領域 $V_l(l = 0, 1, \ldots, P-1)$ を構成し，P 台のプロセッサ $(PE_l : ProcessingElement)$ で並列処理をする手法が考えられる．このときに，次の3種類のデータを通信する必要が生じる．

1. Step 1b の式 (2.9) の計算後に各ボクセル $V(x,y,z)$ について $S(x,y,z)$ の計算の可否情報 (初回のみ).

2. Step 1c の式 (2.10) で $\max_V S(x,y,z), \min_V S(x,y,z)$ の計算に必要な $\max_{V_l} S(x,y,z), \min_{V_l} S(x,y,z)$ の値.

3. Step 1a の式 (2.8) と Step 2 の式 (2.11)，Step 3 で計算に必要な $e(x,y,z)$ の値

1. については，あるボクセル $V(x,y,z)$ から視点 o_n へ投影した時に，画像 I_n 中に矩形領域 $R_n(x,y,z)$ を作れない場合が生じる．この時の $V(x,y,z)$ は SSD を求められないため，$S(x,y,z)$ が求められない．$S(x,y,z)$ の計算の可否は，直線上のボクセルを数え上げる時に計算から省くために必要になる．そのため，各 PE が担当する $V(x,y,z)$ について $S(x,y,z)$ の計算の可否を調べ，それを他の全ての PE へと通信をする．

2. については，$\max_V S(x,y,z), \min_V S(x,y,z)$ の計算のために，各 PE_l に割り当てられたボクセル領域 V_l 内での $\max_{V_l} S(x,y,z), \min_{V_l} S(x,y,z)$ を各 PE それぞれで求め，その値を代表として PE_0 が集めて，PE_0 が $\max_V S(x,y,z), \min_V S(x,y,z)$ の値を求めて各 PE へと返す．

3. については，あるボクセル $V(x,y,z)$ の計算に使用するデータは，図 3.1 のように $\phi_n(x,y,z)$，$\phi'_n(x,y,z)$，$\Phi(x,y,z)$ で示す，カメラの視点 o_n と $V(x,y,z)$ を通る直線 (または線分) 上のボクセルが持つデータである．そのため，あるボクセル $V(x,y,z)$ の計算を担当する PE_l は，ボクセル領域 V_l 内にその直線上のボクセルデータが無ければ，他の $PE(PE_l$以外$)$ から通信して持ってくる必要がある．ボクセル相互の参照関係は各ボクセルで違うため，通信する必要のあるデータは極めて複雑になってしまう．

通信には，以上のような特徴があるので，本研究では，次小節で示す分割方法を提案し通信量の効果的な削減を図る

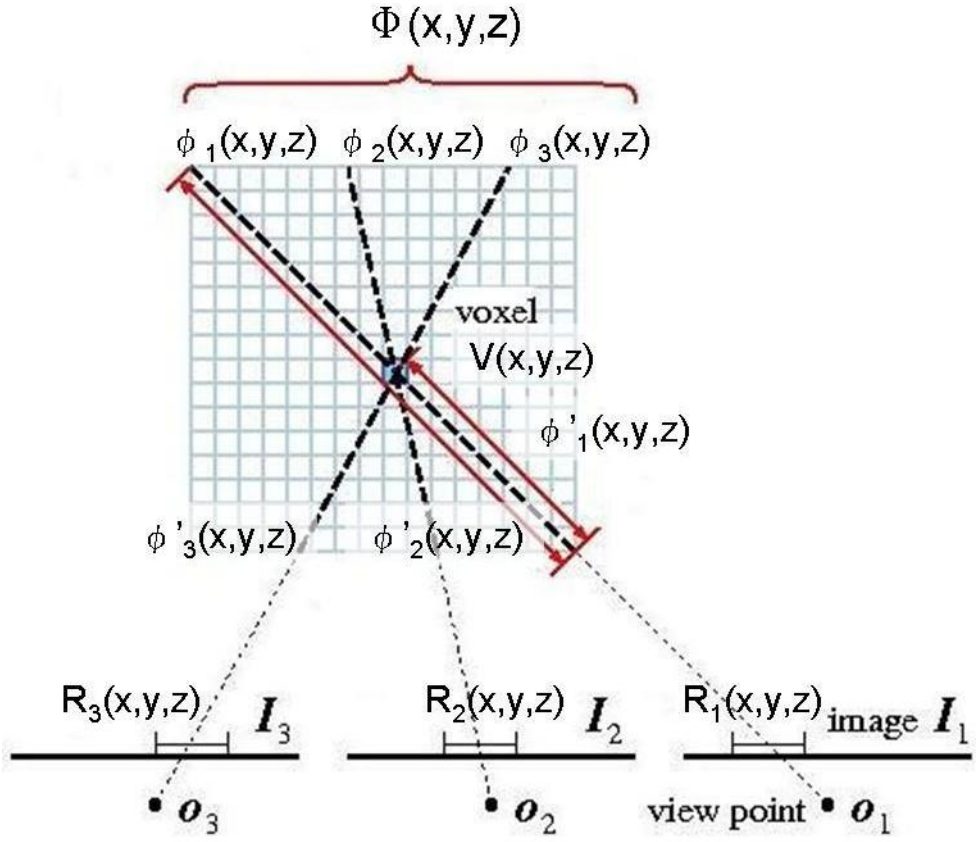


図 3.1: 視点とボクセルの関係（y 軸方向から見た図）

3.2.2 ボクセル空間の分割

ボクセル空間の分割方法としては，空間全体を単純に水平方向に P 分割する手法や，垂直方向に P 分割する手法がある．しかし，あるボクセル $V(x,y,z)$ の計算に使用するデータは，$\phi_n(x,y,z)$，$\phi'_n(x,y,z)$，$\Phi(x,y,z)$ で示されているように，カメラの視点 o_n と $V(x,y,z)$ を通る直線 (または線分) 上のボクセルが持つデータである．そのため，単純に水平，垂直方向に分割した場合，計算に必要なボクセルデータが他の PE に存在する割合が増加し，PE 間の通信量が増加してしまう可能性がある．

このことから，本研究では，カメラの視点と各ボクセルを通る直線の傾きを用いてボクセルを分割する通信量の少ない並列計算法を提案する．具体的には，各カメラの視点 o_n との距離が最小になる直線を求め，その直線を軸とした座標変換をし，その直線上にある視点の平均の座標から各ボクセルを通る直線の傾きを求め，その傾きの大きさの順にソートして，ボクセル全体を P 分割し，P 台の PE で並列処理をする．

以下に，本研究で提案する並列計算法の手順を示す．

1. 各視点 $o_n(x,y,z)$ に配置されるカメラの座標の平均から視点の平均 $o'(x,y,z)$ の座標を次式により求める．

$$o'(x,y,z) = \frac{\sum_{n=1}^{N} o_n(x,y,z)}{N} \tag{3.1}$$

2. 求めた視点の平均 $o'(x,y,z)$ が原点となるように，元の $o_n(x,y,z)$ と $V(x,y,z)$ を $o'_n(x_n,y_n,z_n)$ と $V'(x,y,z)$ へ次式により平行移動する．

$$o'_n(x_n,y_n,z_n) = o_n(x,y,z) - o'(x,y,z) \tag{3.2}$$

$$V'(x,y,z) = V(x,y,z) - o'(x,y,z) \tag{3.3}$$

3. $o'_n(x_n,y_n,z_n)$ との距離が最小となる直線 $at = 0$(原点を通る) を次式により計算する．ただし，$a = (a_x, a_y, a_z)$，t は任意の変数とする．

$$\begin{bmatrix} \sum_{n=1}^{N} x_n x_n - \lambda & \sum_{n=1}^{N} x_n y_n & \sum_{n=1}^{N} x_n z_n \\ \sum_{n=1}^{N} x_n y_n & \sum_{n=1}^{N} y_n y_n - \lambda & \sum_{n=1}^{N} y_n z_n \\ \sum_{n=1}^{N} x_n z_n & \sum_{n=1}^{N} y_n z_n & \sum_{n=1}^{N} z_n z_n - \lambda \end{bmatrix} \begin{bmatrix} a_x \\ a_y \\ a_z \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \tag{3.4}$$

ここで求めた 3 つの固有値を値が大きい順に λ_1，λ_2，λ_3，各々に対応する固有ベクトルを $a1 = (a1_x, a1_y, a1_z)$，$a2 = (a2_x, a2_y, a2_z)$，$a3 = (a3_x, a3_y, a3_z)$ とする．

4. $o'_n(x_n, y_n, z_n)$ との距離が最小となる直線 $a1t$(原点を通る) を X 軸として座標変換し，求めた視点の平均 $o'(x, y, z)$ から全てのボクセルに対して引いた各直線の傾き $g(x, y, z)$ を次式により求める．

$$g(x, y, z) = \frac{Y(x, y, z)}{\sqrt{X(x, y, z)^2 + Z(x, y, z)^2}} \tag{3.5}$$

ただし，$X(x, y, z), Y(x, y, z), Z(x, y, z)$ はそれぞれ視点の平均 $o'(x, y, z)$ から各ボクセル $V(x, y, z)$ までの座標変換した X 軸，Y 軸，Z 軸方向の距離として次式のように定める．

$$X(x, y, z) = V'(x, y, z) \cdot a1(a1_x, a1_y, a1_z) \tag{3.6}$$

$$Z(x, y, z) = V'(x, y, z) \cdot a2(a2_x, a2_y, a2_z) \tag{3.7}$$

$$Y(x, y, z) = V'(x, y, z) \cdot a3(a3_x, a3_y, a3_z) \tag{3.8}$$

5. 求めた傾き $g(x, y, z)$ の大きさの順にソートしてボクセル全体を P 分割し，P 個のボクセル領域 $V_l(l = 0, 1, \ldots, P-1)$ を構成する．この時，$V = V_0 \cup V_1 \cup \ldots \cup V_{P-1}$ であり，各ボクセル領域 V_l は傾き $g(x, y, z)$ の小さい順に V_0 から V_{P-1} まで $\parallel V \parallel /P$ 個のボクセルで構成される．ここで，$\parallel V \parallel$ は全ボクセル V の要素数である．

6. 各ボクセル領域 V_l を PE_l に割り当てて，PE_l は割り当てられたボクセル領域 V_l の $V(x, y, z)$ についてのみ計算を行うことで並列的に処理を行う．

ボクセルを $P = 4$ として分割する場合の概略図を図 3.2 に示す．図に示すように，各カメラの視点 o_n との距離が最小になる直線 $a1t$ を求め，その直線を軸とした座標変換をし，その直線上にある視点の平均の座標 $o'(x, y, z)$ から各ボクセルを通る直線の傾き $g(x, y, z)$ を求め，その傾きの大きさの順にソートして，ボクセル全体を P 分割し，P 台の PE で並列処理をする．このとき, 各 PE_l がボクセル領域 V_l の $V(x, y, z)$ について計算をすることで並列計算が行われる．この手法により，同一直線上のボクセルデータが同一 PE 内に比較的多く配置されるため，単純にボクセル全体を水平，垂直平面で P 分割する場合に比べて，各 PE 間のデータ通信量を少なく抑えることが可能になる．

3.3 実験方法

本節では，前節で提案した並列化手法の有効性を検証するため，対象物体を撮影した実画像を用いて，クラスタマシンで計算時間を求める実験を行った結果を示す．

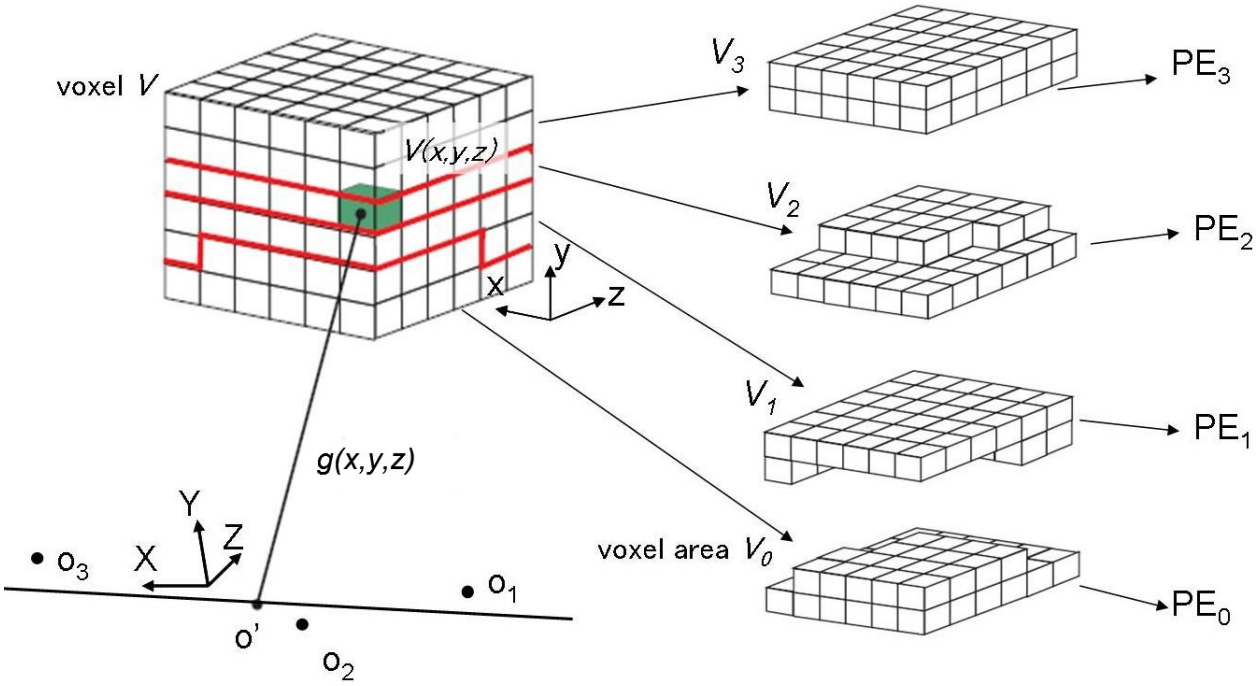


図 3.2: ボクセルを視点の平均からの傾きにより分割し各 PE に割り当て

3.3.1 実験内容

ボクセル全体を水平方向 (x-z 平面に平行な方向) に P 分割して，P 台の PE で並列処理をした場合と，前節で提案した並列化手法を用いて，直線の傾きによりボクセル全体を P 分割して，P 台の PE で並列処理をした場合の計算時間の結果を比較し，評価を行う．

3.3.2 実験環境

実験には 4 台のカメラと 2 個の物体 Object1,2 を用い，図 3.3，3.4 に示すように各々を配置した．各カメラで撮影し実験に用いた画像 I_1〜I_4 を図 3.5 に示す (画像サイズは 680 × 480 画素)．計測範囲である三次元空間は，x, y, z 方向に各々100 × 100 × 100 個のボクセルに分割している (1 ボクセルのサイズは 8 × 2 × 8mm)．また，式 (2.9) 中の SSD を求める際には，並行化した画像対を用い，$R_n(x, y, z)$ として 17 × 17 画素の矩形領域を使用している．

なお，式 (2.11) 中のパラメータを $\alpha = 1.5$ とし，物体境界に対応するボクセルを判定する際の $w(x, y, z)$ に対する閾値は $\gamma = 1.3$ とし，Step 2 の反復回数を 10 回，Step 1, 2 の反復回数を 2 回とした時の計算時間を測定した．

また，実験で用いたクラスタマシンの性能を表 3.1 に示す．クラスタマシンには，

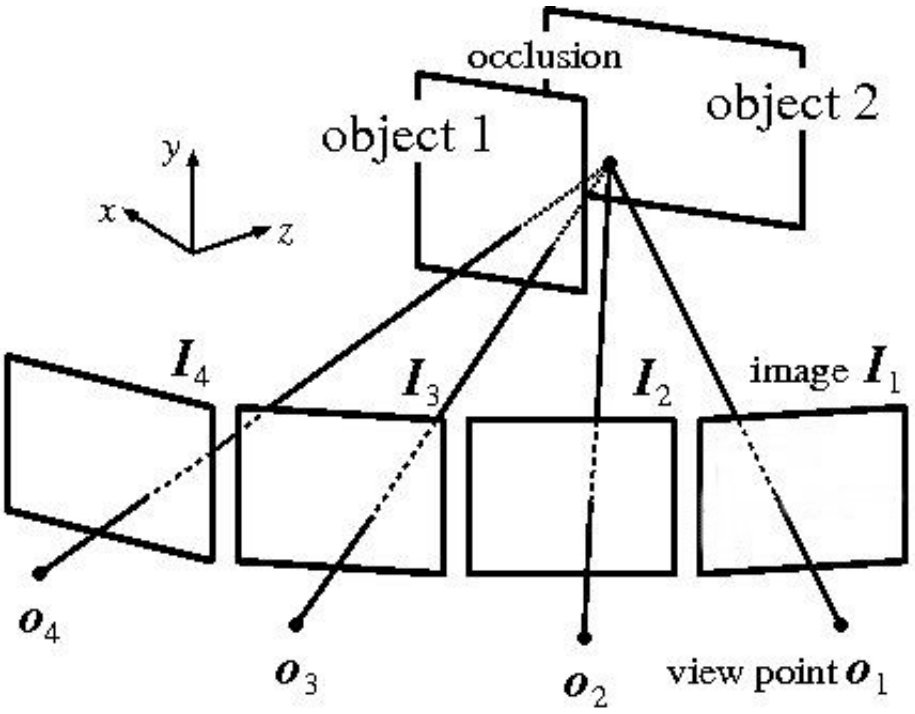


図 3.3: 視点と物体 (object1,2) の配置

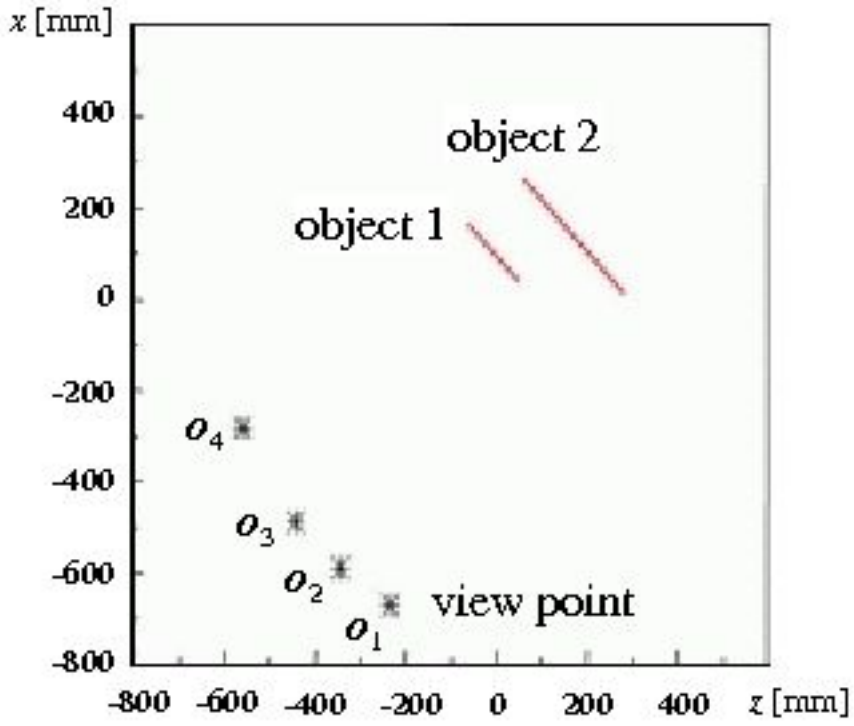


図 3.4: 視点と物体の配置 ($x-z$ 平面へ投影)

MPI ライブラリが提供されているため，本実験では，各ノード間の通信に MPI 関数を用いてデータの通信を行っている．

参考までに，今回用いた三次元情報の整合性を考慮した手法によって求めた三次元形状計測結果を図 3.6 に示す．

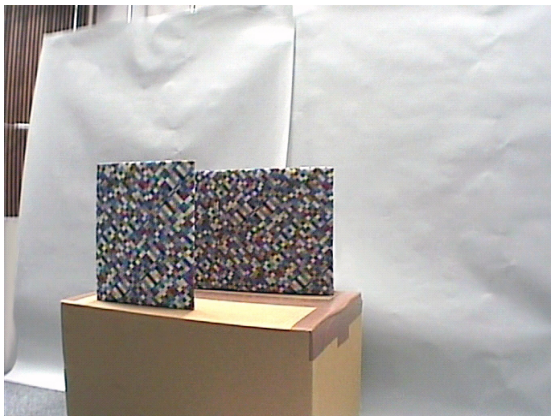
表 3.1: 計算機性能

CPU	AMD Opteron DP Model 246 Processor 2.0GHz
Memory	PC3200 ECC Registered DDR-SDRAM 2GB
Number of Nodes	23
OS	SuSE Linux Enterprise Server 8
Compiler	PGI Compiler
MPI Library	MPICH-SCore

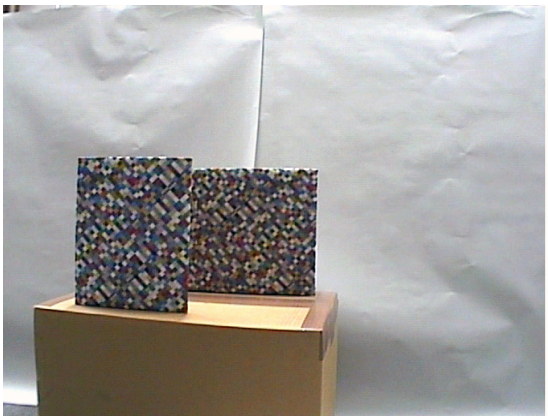
3.4 水平分割と傾きによる分割の計算時間の比較

図 3.7 に，ボクセル V を水平方向に P ($P = \{1, 2, 4, 8, 16\}$) 分割して三次元形状計測を行った際の計算時間の内訳を示す．ここで，横軸は計算に使用したプロセッサ (PE) 数，縦軸は計算に要した時間である．PE 数が 1 の場合は，グラフの下から，初期設定に必要な計算時間，Step 1，Step 2，Step 3 の各計算時間が積み上げられている．また，PE 数が 2 以上の場合には，Step 1 と Step 2 の計算時間の間に Step 1 の通信時間が，Step 2 と Step 3 の計算時間の間に Step 2 の通信時間がさらに積み上げられる．

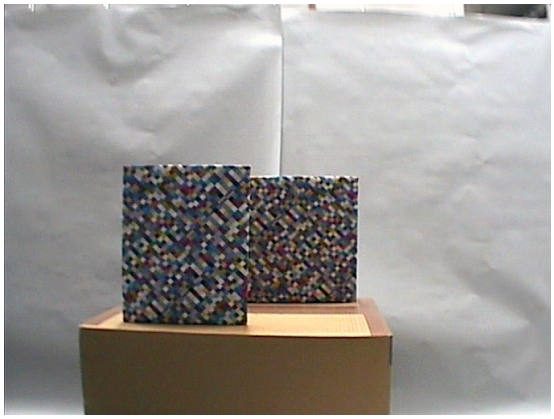
図 3.7 より，水平分割の場合でも，PE 数を増加させるにつれて，各 Step とも計算時間が削減されていることがわかる．これは，クラスタマシンを用いて並列計算をするとき，各 PE がそれぞれの担当ボクセル領域内に存在するボクセルの計算だけを行うためである．特に，$e(x, y, z)$ の値を繰り返し更新する Step 2 の計算時間の削減効果が大きい．しかし，PE 数を増加させるにつれて，各 PE の計算に必要なボクセルデータが他の PE に存在する割合が増加し，PE 間の通信量が増加するため，通信時間は増加してしまっている．そのため，ボクセル V を水平方向に P 分割したときの全体の計算時間は，PE 数が 8 前後で頭打ちになってしまう．また，PE 数が 1 の時と 2 の時の Step 1 や Step2 の計算時間を比較すると，PE 数が 2 倍となるのに計算時間の減りが少ない．これは，PE 数が 2 以上の時には，通信が必要なボクセルを数え上げる処理が新たに必要になるからである．なお，初期設定に必要な計算時間は，各 PE が同様な計算を個別に行う必要があり並列化できない処理で，PE 数が増加しても計算時間に変化が生じない．



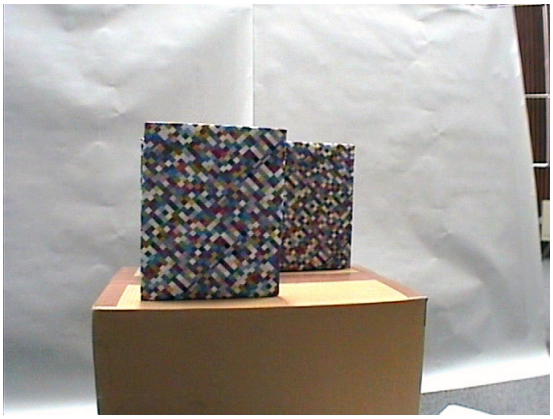
I_1



I_2



I_3



I_4

図 3.5: 実験に用いた入力画像

図 3.6: 三次元形状計測結果

図 3.8 に，提案した並列計算法によりボクセル V を P 分割して三次元形状計測を行った際の計算時間の内訳を示す．水平方向に P 分割した場合と同様に，各 PE がそれぞれの担当ボクセル領域内に存在するボクセルの計算だけを行うため，各 Step とも計算時間が削減されていることがわかる．あるボクセル $V(x,y,z)$ の計算に使用するデータは，カメラの視点 o_n と $V(x,y,z)$ を通る直線 (または線分) 上のボクセルが持つデータであるため，視点の平均 $o'(x,y,z)$ から $V(x,y,z)$ に引いた直線の傾き $g(x,y,z)$ によってボクセル V を分割することで，各 PE の計算に必要なボクセルデータが他の PE に存在する割合が低下し，PE 間の通信量が低下するため，通信時間が削減できたことがわかる．

図 3.9 に，水平分割をした場合の速度向上比と，傾きによる分割を行った場合の速度向上比を示す．なお，速度向上比は次式で定義される指標である．

$$\text{速度向上比} = \frac{P = 1 \text{ の時の計算時間}}{P = \{1,2,4,8,16\} \text{ の時の計算時間}} \tag{3.9}$$

図 3.9 より，水平分割を行った並列計算では PE 数が 16 の時に約 2.2 倍の速度向上であったのに対して，傾きによる分割を行った並列計算において，PE 数が 16 の時に約 6.7 倍の速度向上を達成することがわかる．

3.5 まとめ

本章では，ボクセルベースの三次元形状計測手法に対して，三次元情報の整合性を考慮した手法を例に用いることで，計算時間を削減するための並列化手法を提案した．提案した並列化手法では，複数のカメラの視点から，各視点との距離が最小になる直線と視点の平均を求め，その直線を軸として視点の平均とボクセルを通る直線の傾きを求め，その傾きの大きさの順にソートしてボクセル空間を分割する．これにより，並列化の際に生じる通信量の増大という問題に対処し，三次元形状計測手法を高速化する．そして，その有効性を示すための評価実験を行い，結果を示した．

次章では，この並列化手法を改善するための通信方法，負荷分散について考察し，その有効性を示すための評価を行う．

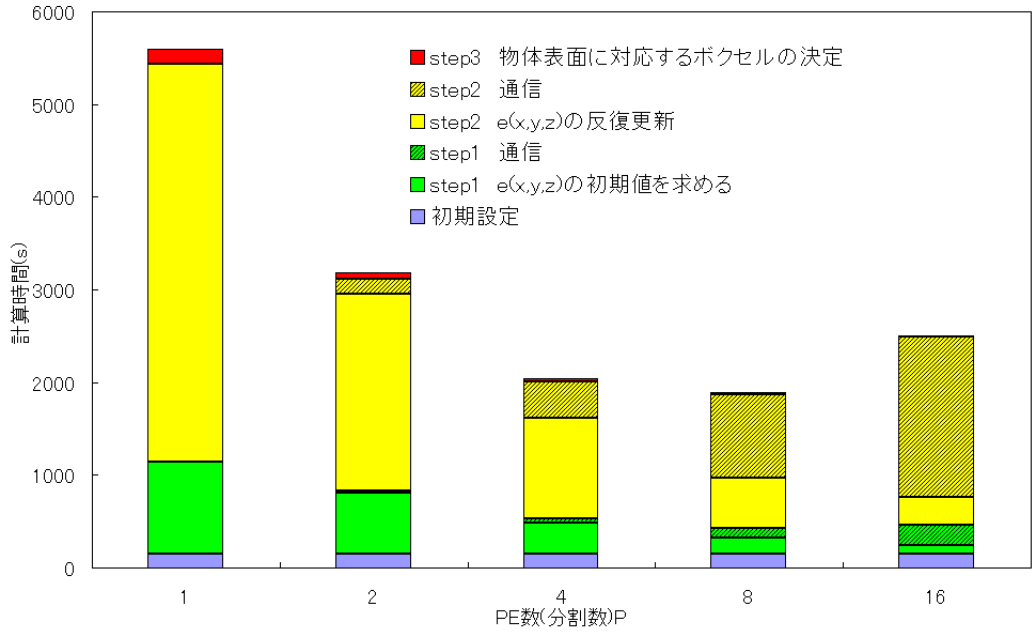


図 3.7: 全体の計算時間 (水平分割)

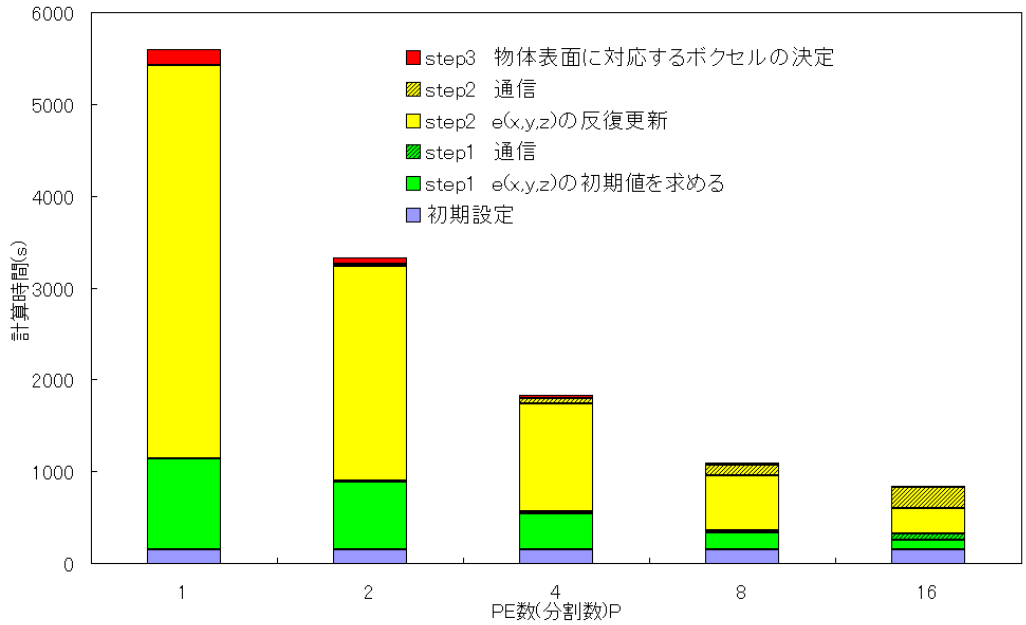


図 3.8: 全体の計算時間 (傾きによる分割)

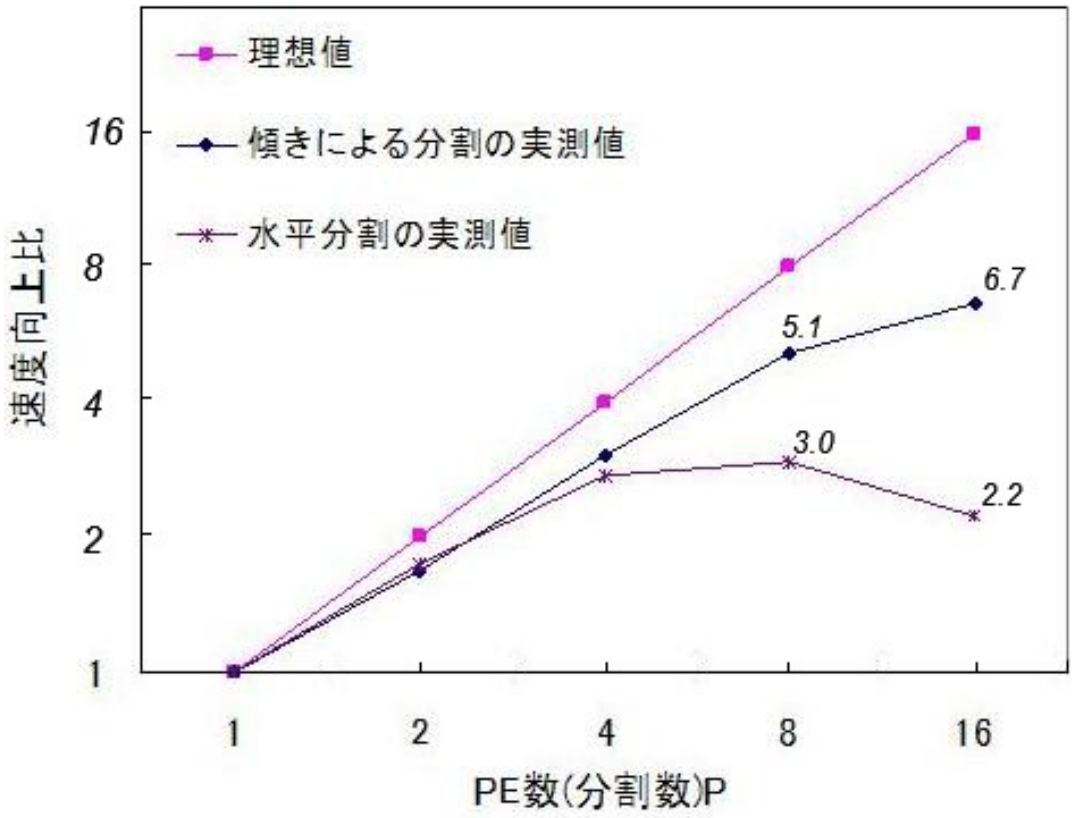

理想値
傾きによる分割の実測値
水平分割の実測値
速度向上比
16
8
4
2
1
6.7
5.1
3.0
2.2
1
2
4
8
16
PE数(分割数)P

図 3.9: 速度向上比

第4章　並列化手法の通信方法と計算負荷に対する考察

4.1　はじめに

第 4 章では，第 3.2 節で提案した並列化手法を改善するための通信方法や負荷分散について検討し，その有効性を示すための評価実験を行い，実験結果を示し考察を行う．そして，並列化手法に対する考察を行う．

4.2　通信方法に対する考察

4.2.1　逐次通信と並列通信

ボクセル空間を P 分割した時，あるプロセッサ PE_l で計算に必要になるボクセルデータは，PE_l 自身と，接するプロセッサ PE_{l+1} と PE_{l-1} に集中して存在することになる．そのため，PE_l が担当する計算を実行するには隣接する PE 間の通信が発生する．P 台の PE で並列処理を行う場合，以下に示す通り $2P-2$ 回の逐次通信を行う必要がある．ここで矢印はボクセルデータの通信の流れを示す．

1. $PE_0 \rightarrow PE_1$
2. $PE_1 \rightarrow PE_0$
3. $PE_1 \rightarrow PE_2$
4. $PE_2 \rightarrow PE_1$

$\vdots$

$(2P-3)$. $PE_{P-2} \rightarrow PE_{P-1}$

$(2P-2)$. $PE_{P-1} \rightarrow PE_{P-2}$

図 4.1 に，$P = 8$ の場合に逐次通信を行った時のタイムチャートを示す．縦軸は通信をする時の送信元 PE であり，四角の中の PE は送信先 PE を表している．図 4.1 に示されるように，PE_0 と PE_1 で通信を行った後に PE_1 と PE_2 で通信を

行う，という順番で通信を繰り返す．しかし，このような通信方法だと，例えば，PE_0 と PE_1 との間で通信を行っている時は PE_2〜PE_7 がアイドル状態となっているため，通信待ち時間の無駄が生じる．そのため，通信待ち時間を削減するために，本研究では，以下のような順番で並列通信を行う．

1. $PE_0 \rightarrow PE_1 \quad PE_2 \rightarrow PE_3 \quad \cdots \quad PE_{P-2} \rightarrow PE_{P-1}$

2. $PE_0 \leftarrow PE_1 \quad PE_2 \leftarrow PE_3 \quad \cdots \quad PE_{P-2} \leftarrow PE_{P-1}$

3. $PE_1 \rightarrow PE_2 \quad PE_3 \rightarrow PE_4 \quad \cdots \quad PE_{P-3} \rightarrow PE_{P-2}$

4. $PE_1 \leftarrow PE_2 \quad PE_3 \leftarrow PE_4 \quad \cdots \quad PE_{P-3} \leftarrow PE_{P-2}$

図 4.2 に，$P = 8$ の場合に並列通信を行った時のタイムチャートを示す．各 PE がなるべくアイドル状態にならないように並列通信を行うことで通信に必要な時間を削減することができる．特に，$P \geq 3$ の時には，P の値に関わらず 1.〜4. までの 4 段階で隣り合った PE 間の通信を完了させることができるため，P の値が大きいほど通信時間の短縮効果が大きくなる．

なお，P の値を増やしていくにつれて，PE_l の計算に必要なボクセルデータが PE_{l+2} や PE_{l-2}，もしくはそれ以上離れた PE に存在するようになる．しかし，PE_l で必要な大半のボクセルデータが PE_{l+1} と PE_{l-1} に存在するため，PE_{l+2} や PE_{l-2} などとの通信量は非常に少ない．今回は，2 つ以上離れた PE に必要なボクセルデータが存在した場合には，隣接 PE 間での通信が終了した後に，適宜，通信を行うものとした．

4.2.2 通信方法考察後の計算時間の評価結果

並列通信の効果を検証するために，計算時間による評価を行った．ここでの実験環境は第 3.3，3.4 節で評価実験を行ったときと同じ環境で評価をしている．

図 4.3 に，第 3.2 節で示した傾き $g(x, y, z)$ を用いたボクセル空間分割を行い，さらに，並列通信を行って三次元形状計測をした際の計算時間の内訳を示す．加えて，並列通信を行う場合と行わない場合の通信時間のみでの比較を図 4.4 に示す．PE 数が増加するにつれて，PE 間でのボクセルデータの通信量が増加するために，通信時間が増加する傾向がある．しかし，並列通信を行うことで，行わない場合に比べて，通信時間を削減できていることがわかる．PE 数が 2 のときは，2 つの PE 相互で通信をするため並列通信による削減の効果は得られないが，$PE \geq 4$ のときは，PE 数が増加するに従って，並列通信の削減の効果がより大きく得られていることが確認できる．これにより，通信待ち時間の無駄が省かれるため速度向上比が向上し，図 4.5 に示すように PE 数が 16 のときに速度向上比が 8.7 と改善された．

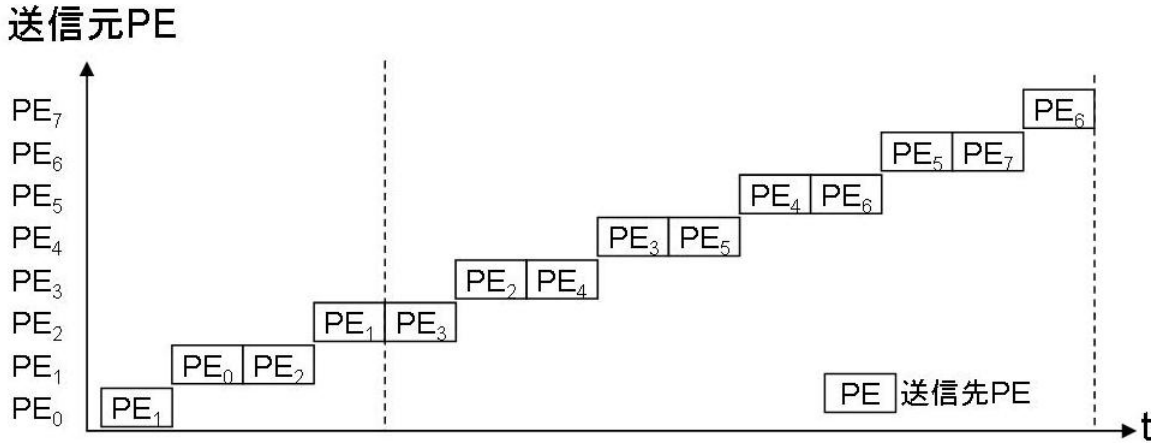


図 4.1: 逐次通信する場合 ($P = 8$)

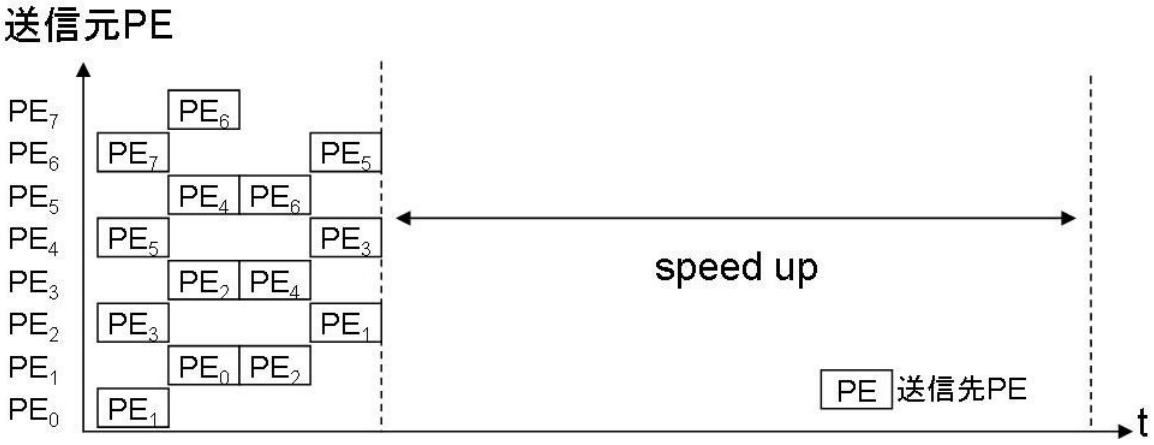


図 4.2: 並列通信する場合 ($P = 8$)

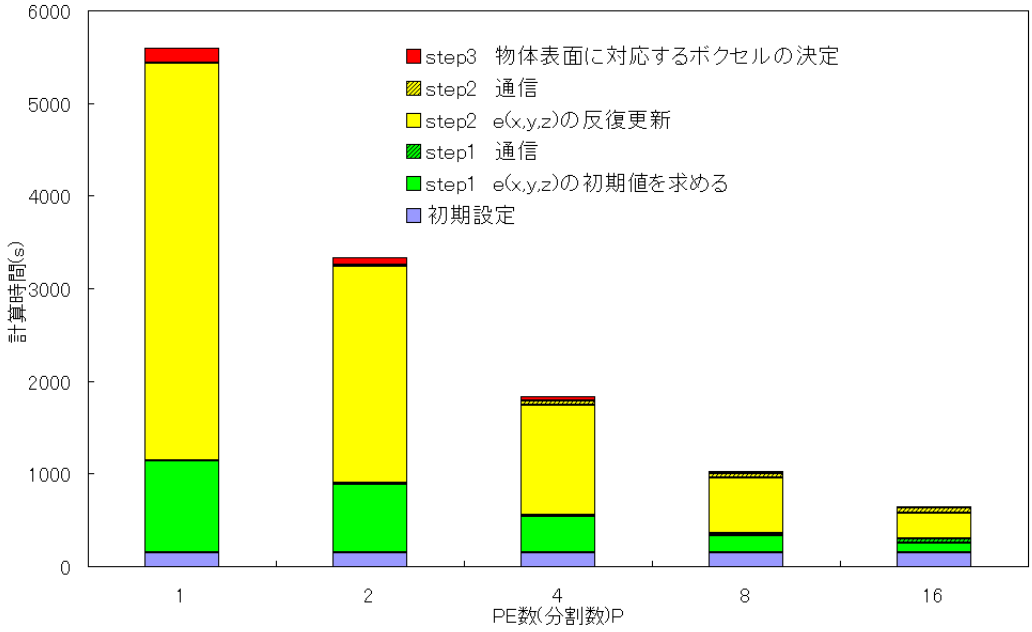


図 4.3: 全体の計算時間 (傾きによる分割 & 並列通信)

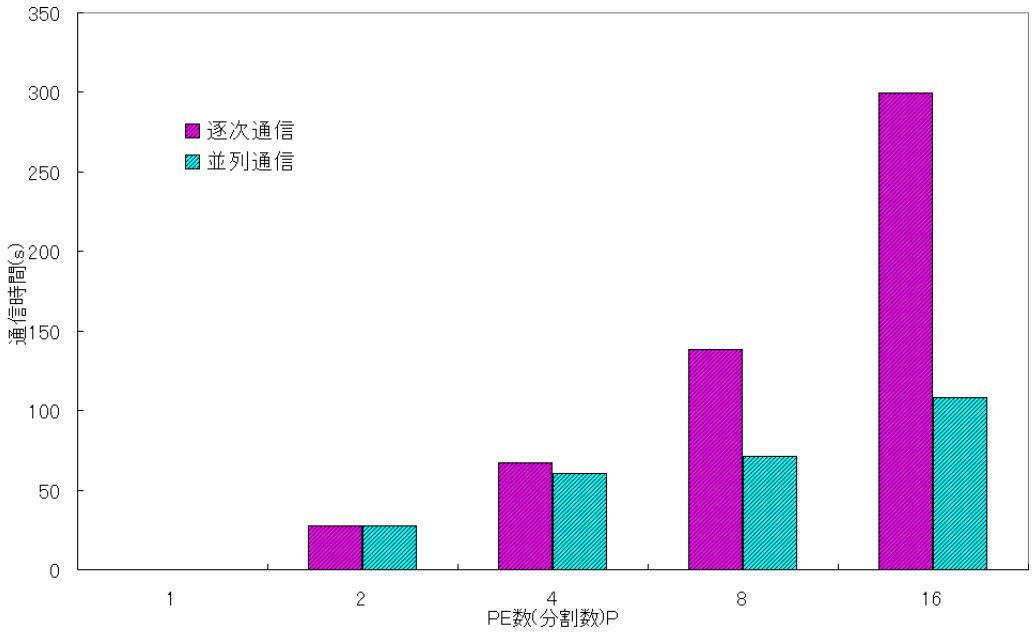


図 4.4: 通信時間の比較

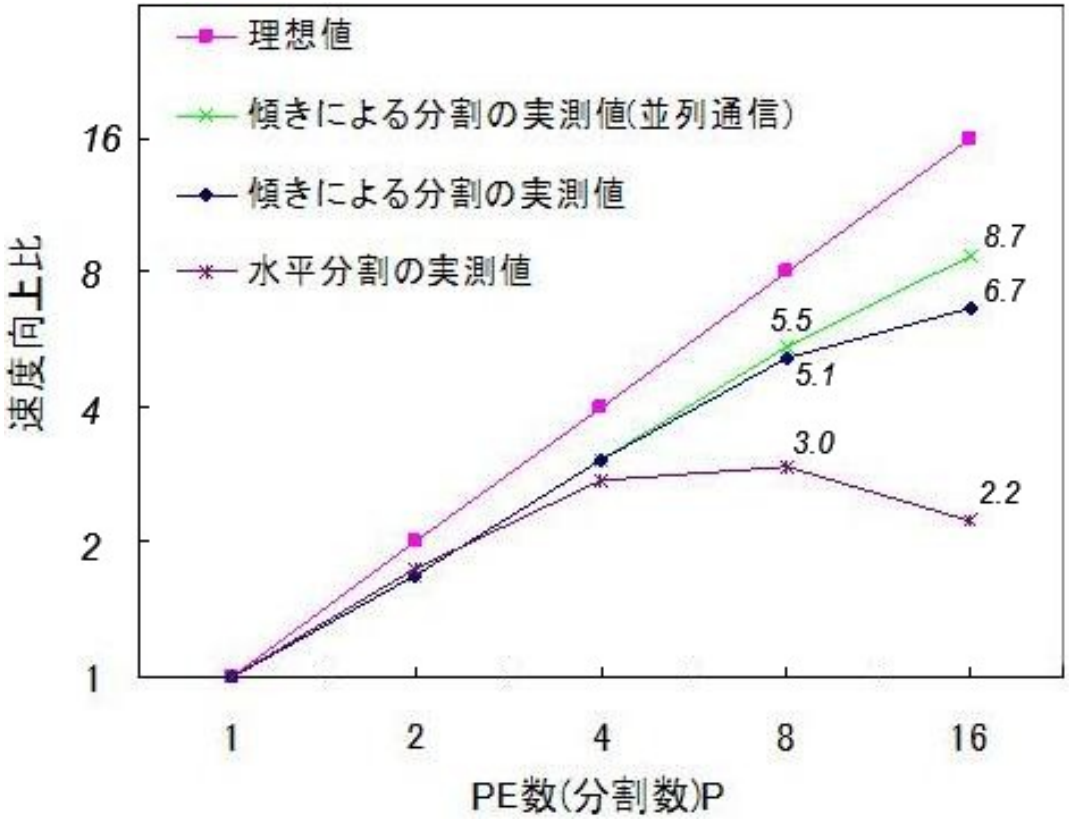

理想値
傾きによる分割の実測値(並列通信)
傾きによる分割の実測値
水平分割の実測値
速度向上比
16
8
4
2
1
8.7
6.7
5.5
5.1
3.0
2.2
1
2
4
8
16
PE数(分割数)P

図 4.5: 速度向上比

4.3 計算負荷に対する考察

4.3.1 負荷分散

三次元情報の整合性を考慮した手法のようなボクセルベースの多眼ステレオ法では，計測対象の周りにボクセルを定義するが，場合によっては定義しただけで計算を行わないボクセル $V(x,y,z)$ が存在することがある．例えば，Step 1b の式 (2.9) では，全ての画像対について，ボクセル $V(x,y,z)$ を I_n へ投影した箇所を中心とする矩形領域 $R_m(x,y,z), R_n(x,y,z)$ 間の画素値の差の 2 乗和である $\mathrm{SSD}_{m,n}(x,y,z)$ を求め，画素の一致度 $S(x,y,z)$ を計算するが，この画素の一致度 $S(x,y,z)$ が計算できない $V(x,y,z)$ が生じることがある．図 4.6 に示すような場合は，ボクセル $V(x,y,z)$ を I_n へ投影した箇所を中心とする矩形領域 $R_m(x,y,z), R_n(x,y,z)$ を作成できるため，全ての画像対について $\mathrm{SSD}_{m,n}(x,y,z)$ を求め，$S(x,y,z)$ を計算するが，図 4.7 に示すように，定義したボクセルの内，端に $V(x,y,z)$ がある場合は，矩形領域 $R_m(x,y,z), R_n(x,y,z)$ を作成できないことがあり，$\mathrm{SSD}_{m,n}(x,y,z)$ が求められず，$S(x,y,z)$ を計算できない．$S(x,y,z)$ を計算できない $V(x,y,z)$ は，式 (2.9) 以降，計算を行わないボクセルとなるため，ボクセル V を傾き $g(x,y,z)$ を用いて P 分割して P 台の PE で並列処理をした場合，上下のボクセル領域を担当する PE_0 や PE_{P-1} は計算を行わないボクセル $V(x,y,z)$ を比較的多く含み，他のボクセル領域 PE_l よりも，計算が速く終了してしまう．そのため，各 PE の内で計算が比較的速く終了する PE と計算が比較的遅く終了する PE が存在してしまうため，計算負荷にバラつきが生じてしまう．

この計算負荷のバラつきを抑えて負荷分散を行うために，ボクセル空間の再分割を行う．Step 1b の式 (2.9) で画素の一致度 $S(x,y,z)$ を計算終了後，各 PE は，自身の担当するボクセル領域 V_l 内のボクセル $V(x,y,z)$ についての $S(x,y,z)$ の計算の可否情報を他の PE へと通信を行う．これにより，どの PE も全ボクセル V の $V(x,y,z)$ について $S(x,y,z)$ の計算の可否情報を保持する．ここで，$S(x,y,z)$ を計算できた全ての $V(x,y,z)$ について式 (3.5) で求めた傾き $g(x,y,z)$ の大きさの順にソートして P 分割し，P 個のボクセル領域 V_l を構成する．各ボクセル領域 V_l を PE_l に割り当てて，PE_l は割り当てられたボクセル領域 V_l の $V(x,y,z)$ についてのみ計算を行うことで並列的に処理を行う．このようにボクセル空間の再分割をすることで，$S(x,y,z)$ を計算できた $V(x,y,z)$ が各ボクセル領域 V_l に均等に配分されて，計算負荷のバラつきを軽減することができる．なお，$S(x,y,z)$ を計算できなかった $V(x,y,z)$ は，そのまま再分割前と同じボクセル領域 V_l が担当する．

4.3.2 計算負荷考察後の計算時間の評価結果

ボクセル空間の再分割による負荷分散の効果を検証するために，計算時間による評価を行った．ここでの実験環境も第 3.3，3.4 節で評価実験を行ったときと同

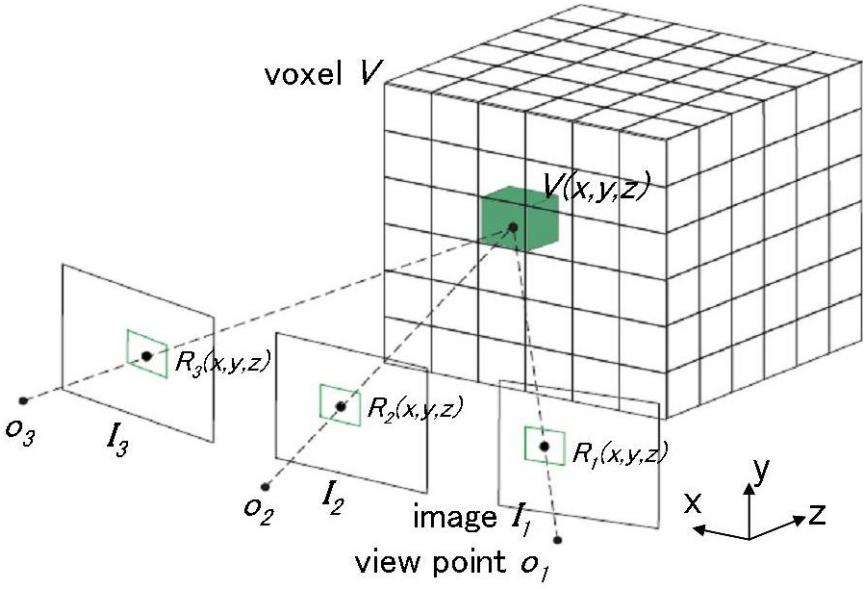


図 4.6: SSD が計算可能なボクセル $V(x, y, z)$ の図

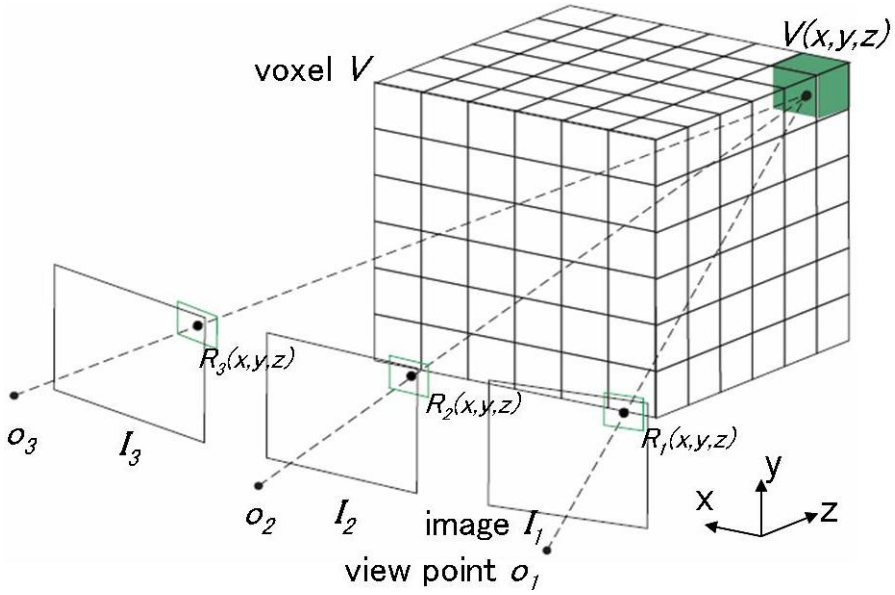


図 4.7: SSD が計算不可能なボクセル $V(x, y, z)$ の図

じ環境で評価をしている．

図 4.8 に，第 3.2 節で示した傾き $g(x, y, z)$ を用いたボクセル空間分割と，第 4.2 節で示した並列通信を行い，さらに，ボクセル空間の再分割を行って三次元形状計測をした際の計算時間の内訳を示す．さらに，ボクセル空間の再分割を行わない場合の計算時間から，再分割を行う場合の計算時間を引いたときの計算時間の削減量を図 4.9 に示す．図 4.9 を見ると，ボクセル空間の再分割を行うことで各 PE に計算が比較的均等に割り当てられることにより計算時間が削減できていることがわかる．PE 数が 2 のときに最も効果的に計算時間の削減をすることができた．これは，ボクセル空間の再分割前に，各 PE の担当するボクセル領域の SSD を計算できるボクセル数のバラつきが大きかったことに起因する．また，PE 数が増加するに従って，計算時間の削減量が少なくなっていくことが確認できる．これは，PE 数が増加することで分割数が増えるため，各 PE の計算負荷の差が小さくなるためである．ボクセル空間の再分割を行うことにより，負荷分散をすることができるため速度向上比が向上し，図 4.10 に示すように PE 数が 16 のときに速度向上比が 8.9 と改善された．

4.4 提案した並列化手法に対する考察

本研究で提案した並列化手法の利点として，スケーラビリティを持つ点が挙げられる．それを示すために，計測範囲である三次元空間を，x, y, z 方向に各々200 × 200 × 200 個のボクセルに分割し (1 ボクセルのサイズは 4 × 1 × 4mm)，定義するボクセルの数を増やすことで三次元形状計測の精度を上げた場合の計算時間の測定を行う．このとき，その他の実験環境は第 3.3 節と同様の条件で測定を行った．図 4.11〜4.14 に，ボクセル V を水平方向に P 分割して三次元形状計測を行った際の計算時間と，第 3.2 節で示した傾き $g(x, y, z)$ を用いたボクセル空間分割を行った際の計算時間と，第 4.2 節で示した並列通信を行った際の計算時間と，さらに，第 4.3 節で示したボクセル空間の再分割を行った際の計算時間を示す．また，図 4.15，4.16 に並列通信をした場合としない場合の通信時間のみでの比較と，ボクセル空間の再分割をした場合としない場合の計算時間の削減量を示す．最後に，図 4.17 にそれぞれの速度向上比を示す．

図 4.11〜4.14 に示されるように，ボクセルの数を 100 × 100 × 100 から 200 × 200 × 200 に増やしたことによりさらに膨大な計算時間が必要になることがわかる．しかし，図 4.17 に示されるようにボクセルの数が 100 × 100 × 100 のときと比べて速度向上比は全体的に向上していることがわかる．この要因としては以下のようなことが考えられる．

1. 計算時間の増加率に対する通信時間の増加率が相対的に少ない．

2. 初期設定に必要な計算時間がボクセルの数に左右されない．

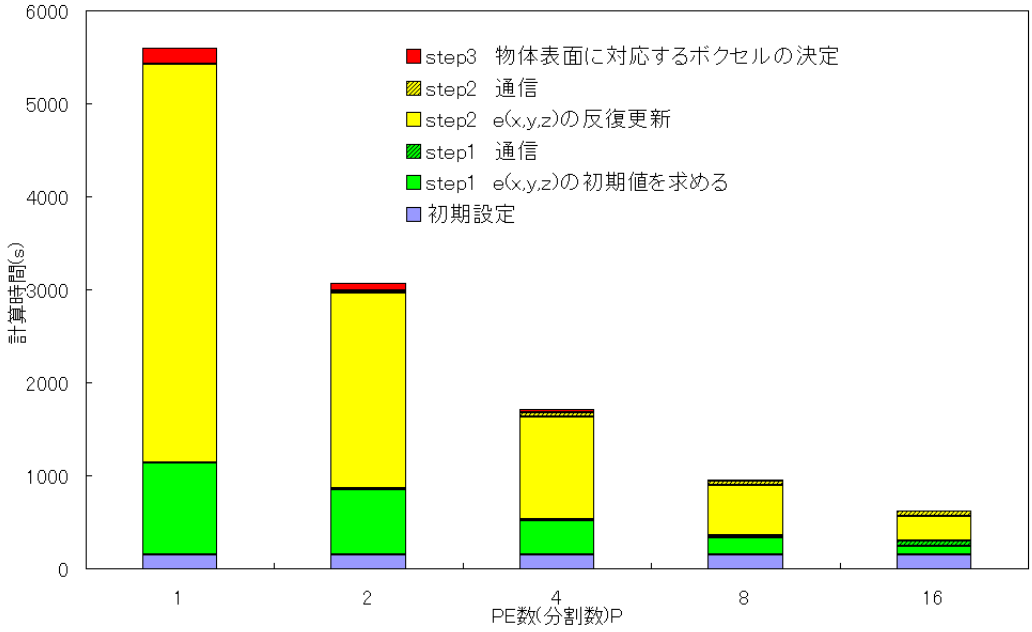


図 4.8: 全体の計算時間 (傾きによる分割 & 並列通信 & 再分割)

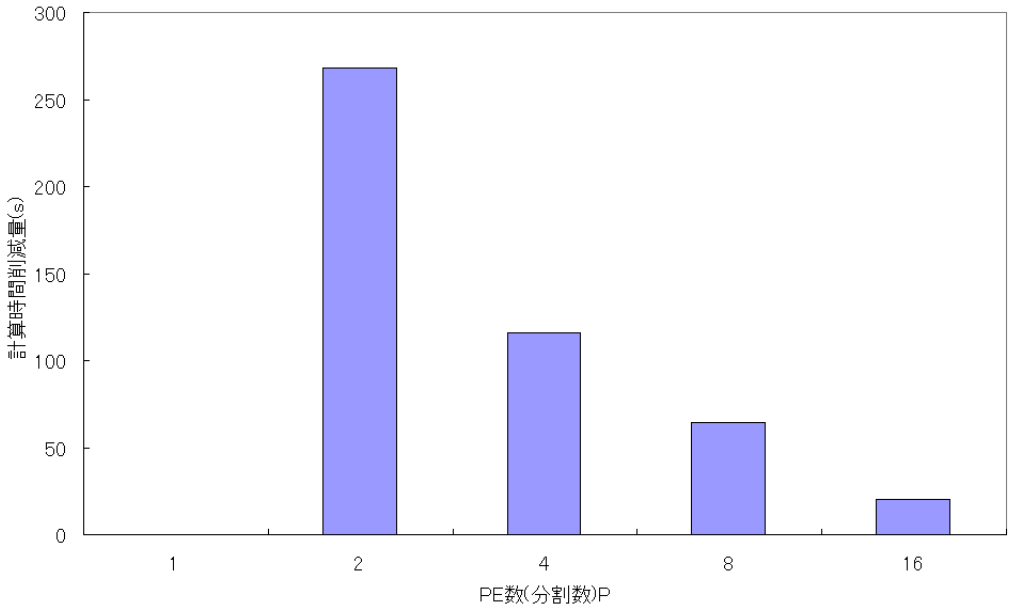


図 4.9: ボクセル空間の再分割による計算時間の削減量

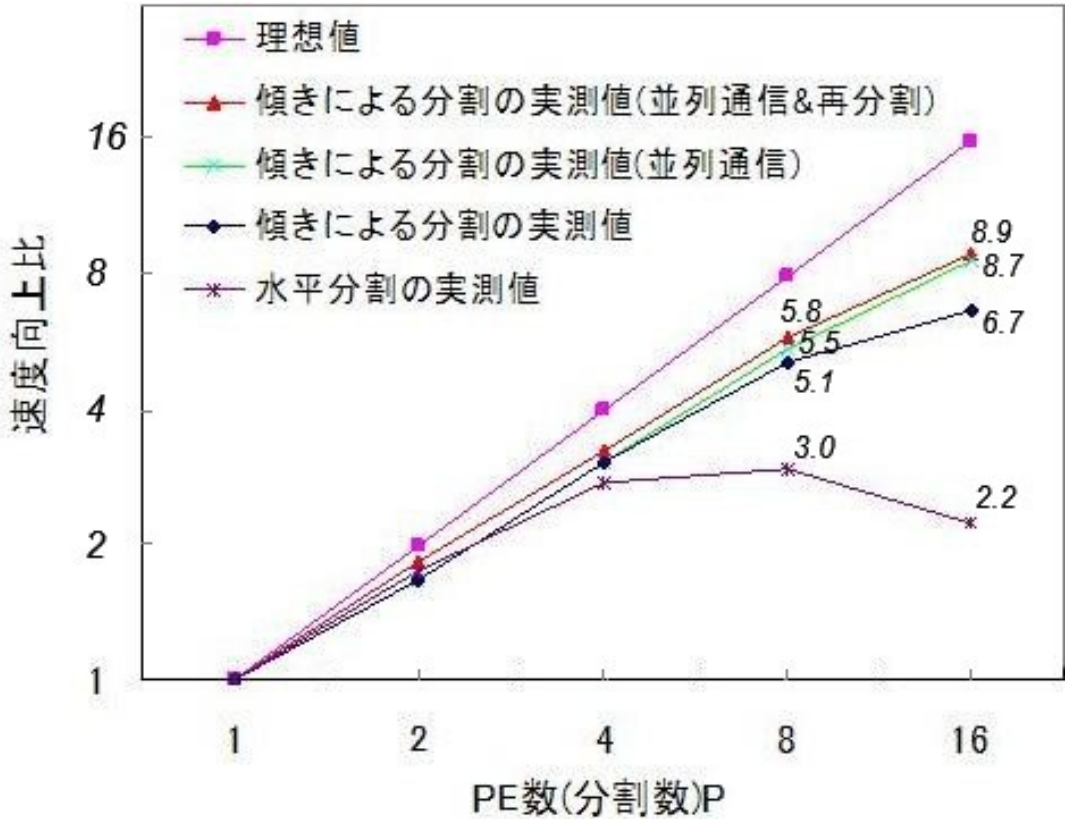

理想値
傾きによる分割の実測値(並列通信&再分割)
傾きによる分割の実測値(並列通信)
傾きによる分割の実測値
水平分割の実測値
速度向上比
16
8
4
2
1
8.9
8.7
6.7
5.8
5.5
5.1
3.0
2.2
1
2
4
8
16
PE数(分割数)P

図 4.10: 速度向上比

1. については，ボクセルの数が増えると計算する必要のあるボクセルが増え，かつ，視点 o_n とボクセル $V(x,y,z)$ を通る直線上にあるボクセルの数も増えるため，必要な計算時間は著しく増加する．しかし，通信時間は，あるボクセル領域 V_l 内の $V(x,y,z)$ について視点 o_n とボクセル $V(x,y,z)$ を通る直線上にあるボクセルを求めたとき，V_l 内に無いボクセルのデータのみを通信すればよいため通信時間の増加は少なくてすむ．よって，通信時間が相対的に少なくなるため速度向上比が向上したと考えられる．

2. については，初期設定に必要な計算時間は，主に画像の分散計算などにかかる時間であるため，カメラの位置，画像，台数によって左右されるが，定義したボクセルの数には左右されない．そのため，ボクセルの数を $200 \times 200 \times 200$ に増やしても初期設定の計算時間はほとんど変化しない．よって，初期設定の計算時間が相対的に少なくなるため速度向上比が向上したと考えられる．

これらの理由により，三次元形状計測をボクセルの数を増やしてより精度よく測定したいという場合には本並列化手法は特に有効となる．しかし，複数のカメラの視点から，各視点との距離が最小になる直線と視点の平均を求め，その直線を軸として視点の平均と各ボクセルを通る直線の傾きを求めるという本並列化手法の性質上，カメラの位置や台数によっては通信量の増加は避けられない可能性がある．特に，カメラの位置をメッシュ状に配置するようなボクセルベースの三次元形状計測手法があった場合には，本並列化手法を上手く適用することが難しいなどの問題点も残る．

4.5 まとめ

本章では，第 3.2 節で提案した並列化手法を改善するための通信方法や負荷分散について検討した．具体的には，送受信を行う PE が重複しないような並列通信を行うことで，通信待ち時間を削減したり，$S(x,y,z)$ を計算できた $V(x,y,z)$ について，ボクセル空間の再分割を行うことで，計算負荷が比較的均等になるようにした．そして，その有効性を示すための評価実験を行い，結果を示した．最後に，提案した並列化手法について考察を行った．

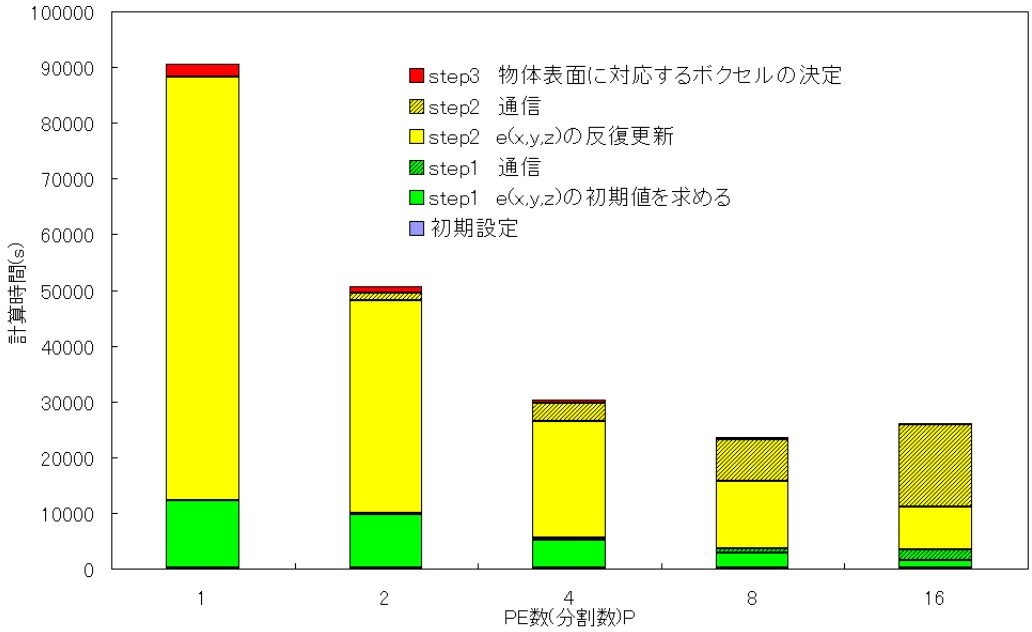


図 4.11: 全体の計算時間 (水平分割)(ボクセル数 200^3)

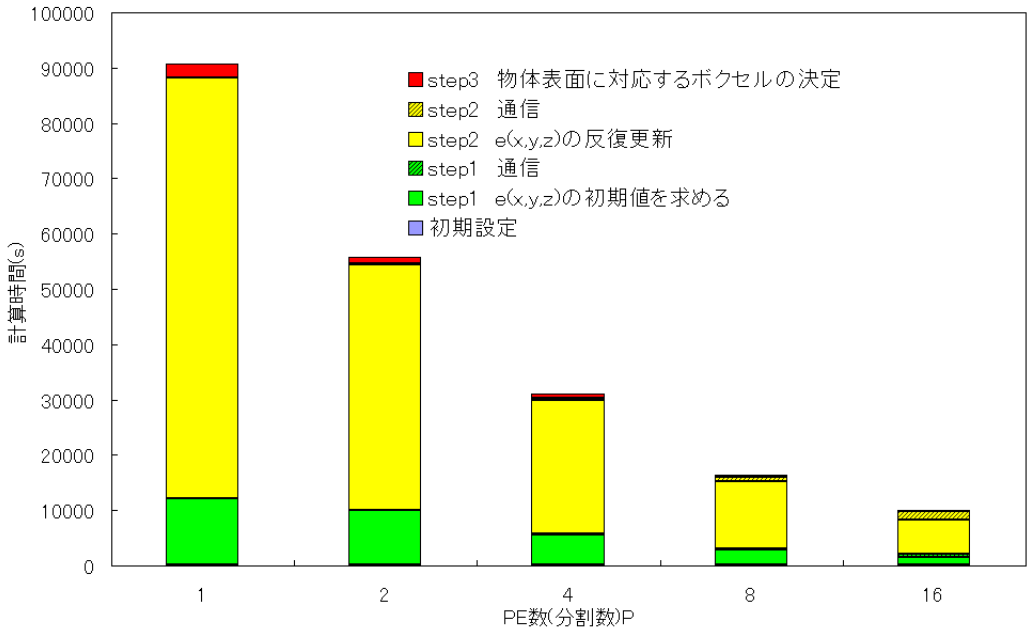


図 4.12: 全体の計算時間 (傾きによる分割)(ボクセル数 200^3)

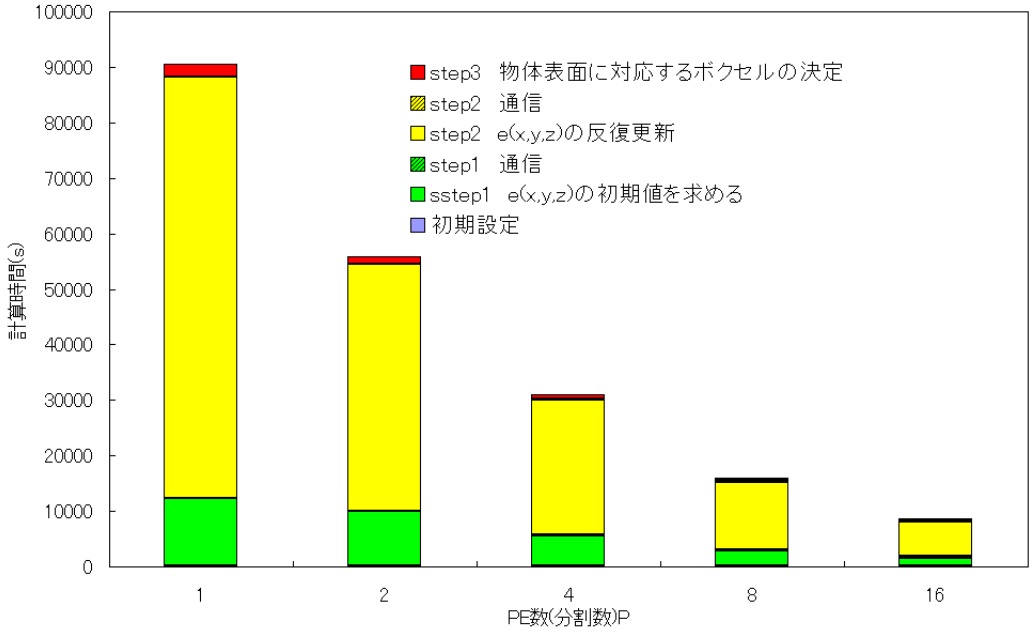


図 4.13: 全体の計算時間 (傾きによる分割 & 並列通信)(ボクセル数 200^3)

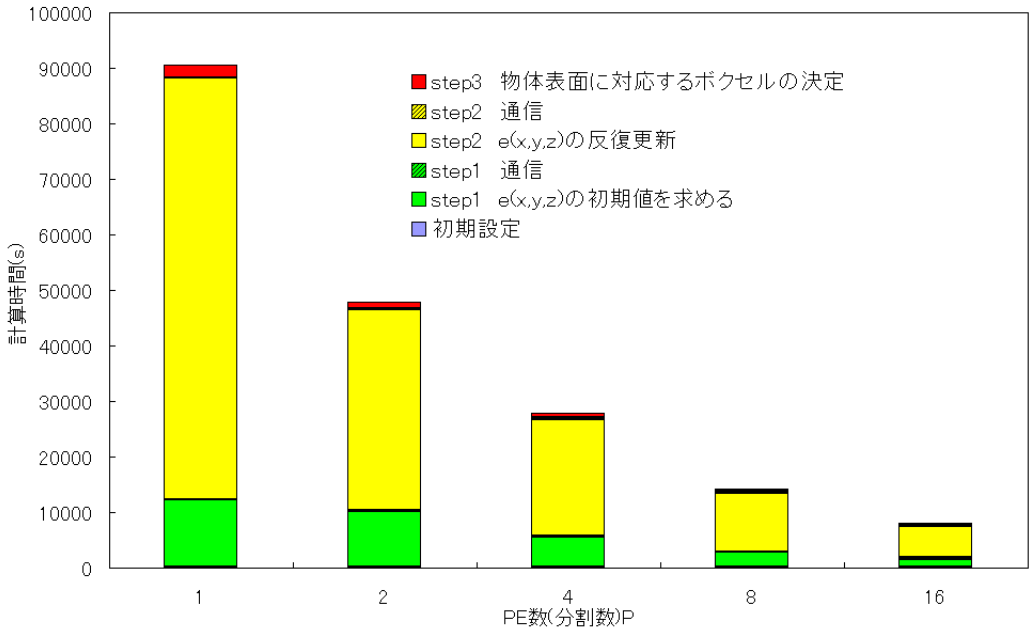


図 4.14: 全体の計算時間 (傾きによる分割 & 並列通信 & 再分割)(ボクセル数 200^3)

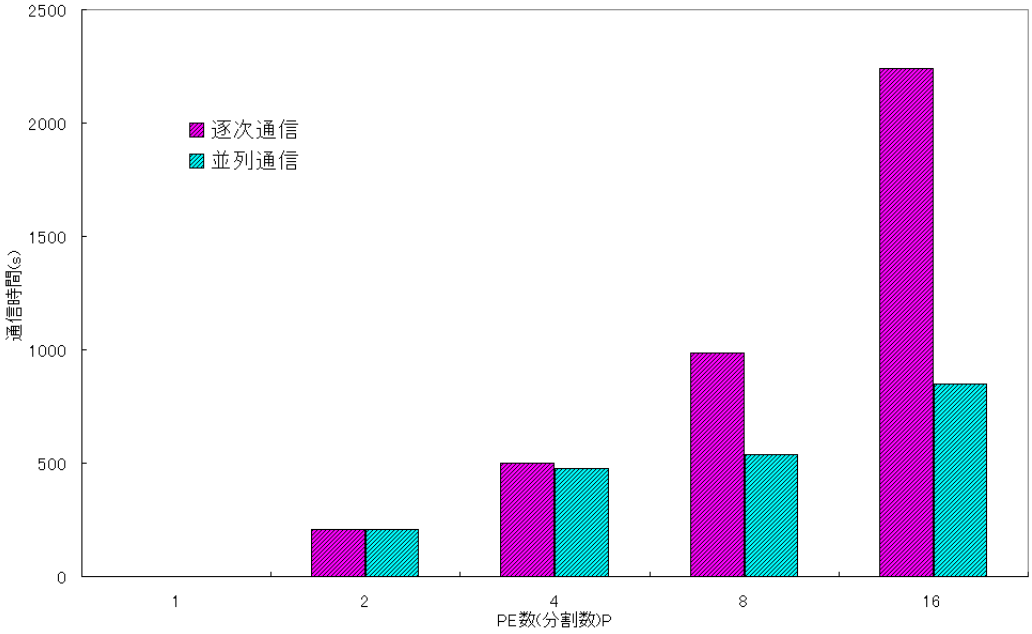


図 4.15: 通信時間の比較 (ボクセル数 200^3)

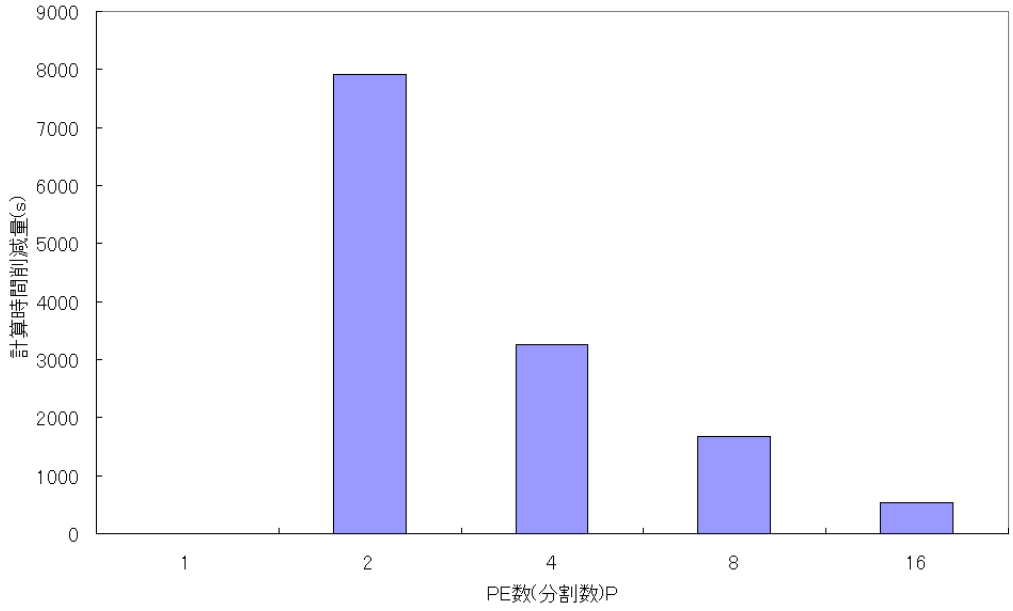


図 4.16: ボクセル空間の再分割による計算時間の削減量 (ボクセル数 200^3)

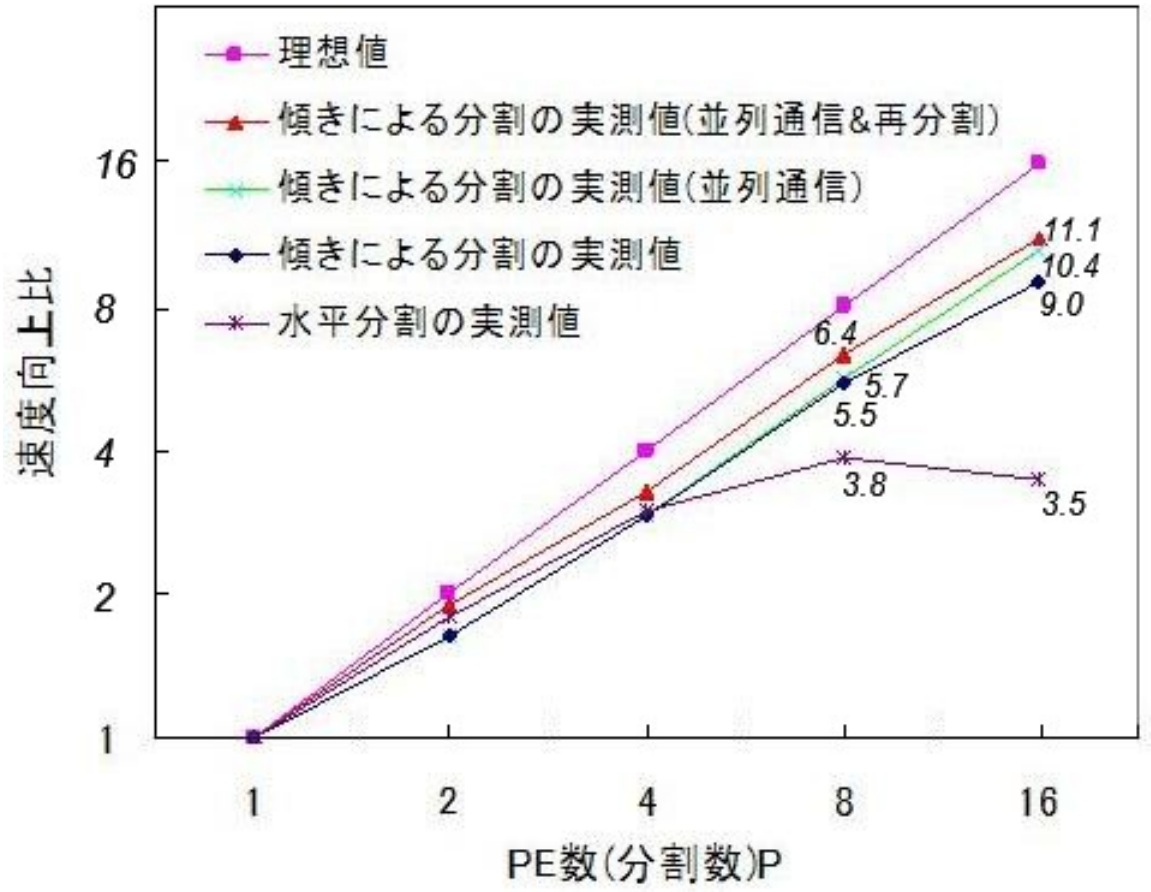


図 4.17: 速度向上比 (ボクセル数 200^3)

第5章　結論

5.1　本研究のまとめ

本論文では，ステレオ法の「遮蔽による誤対応」と,「対応点決定の曖昧さ」の両方の問題を解決する手法として，マルチカメラを用いたボクセルベースの三次元形状計測手法について述べ，その問題点を挙げた．そして，その問題点を解決する手法として，ボクセル空間を視点とボクセルを通る直線の傾きを用いて分割を行うことで通信量を抑えた新たな並列化手法を提案した．そして，その並列化手法を改善するために，通信方法と，計算負荷について考察を行った．

提案した並列化手法では，複数のカメラの視点から，各視点との距離が最小になる直線と視点の平均を求める．そして，その直線を軸として視点の平均と各ボクセルを通る直線の傾き $g(x,y,z)$ を求める．最後に，その傾き $g(x,y,z)$ の大きさの順にソートしてボクセル空間を分割する．これにより，並列化の際に生じる通信量の増加という問題に対処し，三次元形状計測手法の高速化をすることができた．

また，並列化手法を改善するために並列通信と，ボクセル空間の再分割を行った．並列通信は，送受信を行う PE が重複しないように多重通信を行う．これにより，各 PE 間の通信待ち時間を削減することができ，三次元形状計測手法をより高速化することができた．ボクセル空間の再分割は，画素の一致度 $S(x,y,z)$ を計算できた $V(x,y,z)$ を各ボクセル領域 V_l に均等に割り当てることで，各 PE 間の負荷分散を行う．これにより，各 PE 間の計算負荷は比較的均等になり，結果として，三次元形状計測の高速化に繋がった．

5.2　今後の課題

今回，カメラの視点の位置は図 3.3，3.4 に示す 1 通りでしか評価を行っていない．そのため，カメラの視点の位置や台数を変更することで，提案した並列化手法にどのような影響があるか評価実験をする必要がある．また，ボクセル空間の再分割による負荷分散で，$S(x,y,z)$ を計算できた $V(x,y,z)$ を各ボクセル領域 V_l に均等に割り当てても，各 PE_l の計算の終了にある程度の差が見られたため，負荷分散についてはさらに検討する必要がある．

参考文献

[1] 奥富 正敏, “ステレオ視（Stereo Vision）,” コンピュータビジョン：技術評論と将来展望, 松山 隆司, 久野 義徳, 井宮 淳（編）, pp.123–137, 新技術コミュニケーションズ, 1998.

[2] M. Z. Brown, D. Burschka, and G. D. Hager, “Advances in computational stereo,” IEEE Trans. Pattern Anal. Machine Intell., vol.25, no.8, pp.993–1008, 2003.

[3] K.N. Kutulakos and S.M. Seitz, “A theory of shape by space carving,” Int. J. Comput. Vision, vol.38, no.2, pp.199–218, 2000.

[4] 小田倉 聡司, 阿部 亨, 木下 哲男, “三次元情報の整合性を考慮した多眼ステレオ法,” 画像の認識・理解シンポジウム（MIRU2006）論文集，pp.483–488, 2006.

[5] C. Nitschke, A. Nakazawa, and H. Takemura, “Real-time space carving using graphics hardware,” 画像の認識・理解シンポジウム（MIRU2006）論文集，pp.928–933, 2006.

[6] 板野 友哉, 森栄 晃彦, 古川 亮, 川崎 洋, “未観測ボクセルのクラス推定を用いた形状の統合及び補間手法と GPU を用いた高速な実装,” 画像の認識・理解シンポジウム（MIRU2007）論文集，pp.365–371, 2007.

[7] M. Agrawal and L.S. Davis, “A probabilistic framework for surface reconstruction from multiple images,” Proc. 2001 IEEE Comput. Society Conf. Comput. Vision Pattern Recognit., pp.470–476, 2001.

[8] A. Broadhurst, T.W. Drummond, and R. Cipolla, “A probabilistic framework for space carving,” Proc. 8th Int. Conf. Comput. Vision, vol.1, pp.388–393, 2001.

[9] R. Bhotika, D.J. Fleet, and K.N. Kutulakos, “A probabilistic theory of occupancy and emptiness,” Proc. 7th European Conf. Comput. Vision, pp.112–130, 2002.

[10] M. Okutomi and T. Kanade, “A multiple-baseline stereo,” IEEE Trans. Pattern Anal. Machine Intell., vol.15, no.4, pp.353–363, 1993.

[11] S. Roy and I. J. Cox, “A Maximum-Flow Formulation of the N-Camera Stereo Correspondence Problem,” Proc. International Conference on Computer Vision, pp.492-499, 1998.

[12] T. Sato and N. Yokoya, ”New multi-baseline stereo by counting interest points,” Proc. 2nd Canadian Conference on Computer and Robot Vision (CRV2005), pp. 96-103, May 2005.

[13] T. Sato, M. Kanbara and N. Yokoya, “Dense 3-D reconstruction of an outdoor scene by hundreds-baseline stereo using a hand-held video camera,” Int. J. Computer Vision, vol.47, nos.1–3, pp.119–129, 2002.

[14] Y. Nakamura, T. Matsuura, K. Satoh, and Y. Ohta, “Occlusion detectable stereo — Occlusion patterns in camera matrix —,” Proc. IEEE Conf. Computer Vision and Pattern Recognition, pp.371–378, 1996.

[15] C. Silva and J. Santos-Victor, “Intrinsic Images for Dense Stereo Matching with Occlusions,” Proc. European Conf. Computer Vision, pp.100-114,2000.

[16] 延原章平, 和田俊和, 松山隆司, “弾性メッシュモデルを用いた多視点画像からの高精度 3 次元形状復元,” 情報処理学会 CVIM 研究会論文誌, vol.43, SIG11(CVIM5), pp.53-63, 2002.12.

[17] C. L. Zitnick and T. Kanade, “A cooperative algorithm for stereo matching and occlusion detection,” IEEE Trans. Pattern Anal. Machine Intell., vol.22, no.7, pp.675–684, 2000.

謝辞

本論文を終えるにあたり，多大なご指導とともに本研究の機会を与えて下さりました東北大学大学院情報科学研究科堀口進教授に謹んで感謝いたします.

また，本論文の審査をしていただきました東北大学大学院情報科学研究科静谷啓樹教授，並びに，東北大学大学院工学研究科阿曽弘具教授に深く感謝いたします.

本研究を進めるにあたり，有意義なご指導とご助力をして下さいました，東北大学情報シナジー機構阿部亨准教授に心より感謝いたします。

また，常日頃から貴重なご意見・ご助言を頂きました東北大学情報科学研究科福士将助教，東北大学情報科学研究科姜暁鴻准教授に改めてお礼申し上げます。

最後に、研究に限らず多岐に渡ってお世話になりました研究室の皆様，両親共に深くお礼申し上げます。

発表論文

発表論文の一覧

1. “三次元形状計測の並列化手法,” 高橋 一樹, 福士 将, 阿部 亨, 堀口 進, 平成19年度電気関係学会東北支部連合大会講演論文集，p.1E-02, 2007.

2. “多眼ステレオ三次元形状計測の並列計算法,” 高橋 一樹, 福士 将, 阿部 亨, 堀口 進, 電子情報通信学会技術研究報告，Vol.107 No.379，pp.129–134, 2007.