Vol.2014-CVIM-190 No.40
2014/1/24

移動カメラ画像からの３次元形状復元・自己位置推定（SLAM）と高密度な３次元形状復元

関 晃仁[1,a)]

概要：SLAM(Simultaneous Localization and Mapping) は，未知の環境の中でセンサの自己位置と，周囲の環境地図を構築していく技術である．本稿では，SLAM 技術を用いた移動カメラで撮影された画像からの周囲環境の３次元形状復元と，カメラの自己位置推定について，処理の流れに沿って基本的な理論と公開されているソースコードを結びつけて説明する．一度カメラの自己位置が推定できれば，より密度の高い詳細な３次元形状を復元できる．高密度な３次元形状復元を，画像間の対応づけを画像内でローカルに行う方法と画像全体を使う方法の２つに分けて説明する．また，実験に適したカメラとレンズの選定方法について説明する．本稿では，これから３次元形状復元やカメラの自己位置推定を実装する読者がどのような処理が必要なのかを理解し，より深く調べるためのポインタとなることを念頭に説明を進める．

1. はじめに

カメラの自己位置推定は，映像製作や拡張現実 (AR) といった CG の実画像への合わせこみに利用したり [1]，自動車，ロボット等の位置同定や制御に利用したりすることができる [2]．自己位置に加えて周辺環境の３次元形状が把握できることで，障害物を検出して，その位置を知ることもできる [3], [4], [5]．また，密な３次元形状の計測をリアルタイムで行い，福島原子力発電所の廃炉に向けた作業支援に利用する応用例もある [6]．

未知の環境の中で，センサの自己位置と，周囲の環境地図を構築していく技術は SLAM(Simultaneous Localization and Mapping) 技術と呼ばれ，用いる視覚センサによって以下の３つに大別することができる．

- デプスセンサ
- ステレオカメラ
- 単眼カメラ

例えば Microsoft 社の Kinect^{TM}[7] に代表される，近年登場してきたデプスセンサを用いた SLAM 技術が発表されている [8], [9]．デプスセンサでは，画像上の各位置について奥行きを求めることができる．よって，ICP(Iterative Closest Point)[10] を用いて，センサの自己位置を推定することができる．周囲の環境地図は，デプスセンサを用いて生成される．ステレオカメラを用いる場合も，デプスセンサと同じように ICP を用いて自己位置の推定が可能である [11]．

[1] 株式会社 東芝 研究開発センター
[a)] akihito.seki@toshiba.co.jp

１台のカメラで構成される単眼カメラは最もシンプルなシステムであり，例えば Web カメラ等を用いてすぐに実験機材を揃えることができる．ところが単眼カメラは，１枚の画像だけでは何らかの仮定を置かない限り，３次元形状を求めることができない．２枚以上の画像を用いれば，三角測量の原理によって３次元形状を求めることができるが，カメラの自己位置と画像上での対応位置が必要である．本稿では，主に単眼カメラを用いた移動カメラ画像からの自己位置推定と，３次元形状復元について説明する．なお，自己位置にはカメラの３次元位置と向き（姿勢）を含むとして説明する．

SLAM 技術は，さまざまな手法で構成されているが，本稿では細かい理論を説明するのではなく，必要な処理の項目やその考え方，公開されているソースコード [*1] の紹介に絞って説明する．詳細を知りたい読者は，参考文献を調べていただきたい．また，実際に AR，自動車，ロボット等へ利用するためには，リアルタイム性や少ないメモリ使用量を求められる場面が多いと考えられる．本稿では所々で計算時間やメモリ使用量に触れつつ説明する．

1.1 単眼カメラによる自己位置推定と３次元形状復元

単眼カメラを用いたカメラの自己位置推定と３次元形状復元では，一定時間内に得られた運動情報（過去および未来の情報）を一括して処理する手法と，過去に得られた情報を逐次的に用いて，現時点の状態を推定する手法の２通りがある．前者の代表的な手法として，文献 [12], [13] の因子分解法がある．この方法では，画像上から特徴点を複

[*1] 原稿執筆時の 2013 年 12 月時点のリンクである．

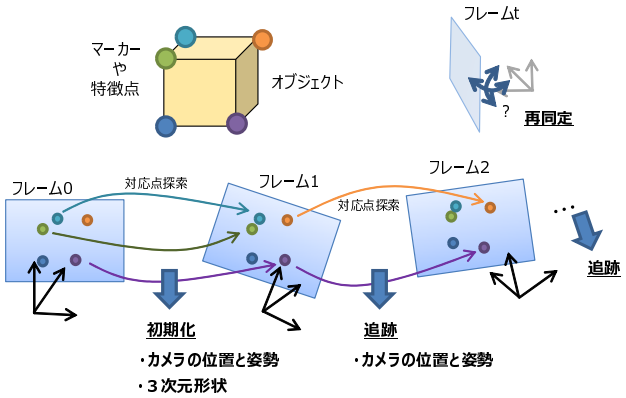

図 1 単眼カメラからの自己位置推定と３次元形状復元の処理概要.

数のフレームにわたって追跡し，追跡された特徴点の位置を'Motion matrix' と'Shape matrix' に分解することで，各時刻におけるカメラの自己位置と剛体の３次元形状を推定している．

後者の手法は，現時点でのカメラの自己位置と周囲の３次元形状がわかるため，オンライン処理が必要なアプリケーションに向く．図１は，カメラの自己位置推定と３次元形状復元の一般的な処理の流れである．単眼カメラでは，最初はカメラの自己位置も周囲の３次元形状も未知であるため，画像に移った２次元情報から処理を始めなくてはならない．

2. 初期化処理

カメラの自己位置と画像に写る特徴点の３次元形状は，相互に関係しており，どちらかが既知であれば他方を推定することができる．本節では，カメラの自己位置と３次元形状が未知である状況下で，最初に行う処理について説明する．

2.1 特定のマーカーを用いた方法

マーカーを用いた初期化方法では，あらかじめマーカーの３次元座標が既知なパタンを環境中に置いておき，マーカーが映るようにカメラを動かす．マーカーの既知な３次元形状からカメラの自己位置を推定することができる (3.1 節)．例えば AR-Toolkit[14] では，四角形状のマーカーを用いている．

マーカーを頼りに初期化を行い，その後に移動しながら入力された画像からマーカーが消えても，特徴点を頼りに 3 節で述べる処理を行いて，自己位置推定ができる [15]．

マーカーの利点として，初期化の成功率が上がること，単眼カメラだけでは求めることのできない物の大きさやカメラの移動量のスケールを一意に決定することができる．

2.2 特定のマーカーを用いない方法

特定のマーカーがない場合には，基本行列 (2.2.2 節)，基礎行列 (2.2.3 節)，射影変換行列 (2.2.1 節) を用いる方法がある．推定に用いる対応点が少ない順に，射影変換行列 (最低 4 点の対応点組)，基本行列 (最低 5 点の対応点組)，基礎行列 (最低 7 点の対応点組) があり，利用できる条件，計算時間，ロバスト性を鑑みて選択するのがよい．複数の対応点組は, 画像間で特徴点の対応付けを行うことで求める．対応づけは，テンプレートマッチング，Lucas-Kanede 法 [16]，SIFT[17] などを用いればよい．実際には，対応付けに失敗した組も含まれているため，RANSAC 等を用いて処理の安定化を図る．RANSAC では，対応点組が少ないほど試行回数を少なくでき，誤対応に対して安定に推定ができる．

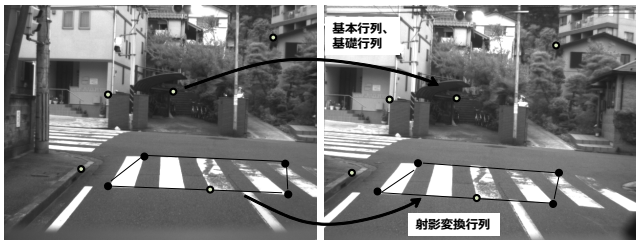

図 2 射影変換行列，基本行列，基礎行列の対応点の位置.

2.2.1 射影変換行列

図 2 の●点のように，３次元空間上の平面が２枚の画像に写っている場合には，射影変換行列 $\mathbf{H}$ を用いて，カメラの自己位置を推定可能である．３行３列の行列で構成される射影変換行列 $\mathbf{H}$ は，式 (1) に示すように平面上にある点の座標 $\tilde{\mathbf{x}}_2$ (ここで，$\tilde{\mathbf{x}}_2 = (u_2, v_2, 1)$ は画像での特徴点の座標を同次座標で表現したものである) を，もう一方の画像の対応位置 $\tilde{\mathbf{x}}_1$ に一意に変換できる．

$$\tilde{\mathbf{x}}_1 = \mathbf{H}\tilde{\mathbf{x}}_2, \quad \mathbf{H} = \mathbf{A}_1 \left(\mathbf{R} + \frac{\mathbf{t}\mathbf{n}^T}{d} \right) \mathbf{A}_2^{-1}. \tag{1}$$

ここで，$\mathbf{A}_*$($*$ は 1 か 2) はカメラの内部パラメータ，$\mathbf{R}$ はカメラ間の回転行列，$\mathbf{t}$ は並進ベクトル，$\mathbf{n}$ は平面の法線ベクトル，d は平面までの距離である．$\mathbf{R}$ と $\mathbf{t}$ は，カメラの自己位置を表している．射影変換行列は，未知数が 8 個 (3 行 3 列目の要素を 1 として固定) あり，画像間の対応点が 1 つあると式が 2 つ立つため，4 つの対応点組があれば線形手法により推定することができる．射影変換行列の推定は，OpenCV[18] に cv::findHomography の関数名で実装されている．

$\mathbf{A}_*$ が既知のとき，すなわちカメラの内部パラメータが既知 [*2] の場合には，射影変換行列を特異値分解することで，カメラの自己位置 ($\mathbf{R}$ や $\mathbf{t}$) や平面の姿勢 ($\mathbf{n}$ や d) を得ることができる [19], [20]．ただし，解は 8 通りの不定性があるため，2 台のカメラから共通の対応点を観察している (すなわち特徴点はカメラより前方に存在している)，2 台のカメラは推定された平面の同じ方向から観測している，等の条件を付けて絞りこむ必要がある．

[*2] 単眼カメラの場合には，各時刻でピントの調整等をしない限りは内部パラメータは不変である．

2.2.2 基本行列

射影変換行列は，対応点が平面上の点である必要があったが，画像中に平面が存在しない場合もあり得る．基本行列や次節で説明する基礎行列では，図 2 の○点のように，空間中に散らばる点の対応関係でよいため，より汎用的である．基本行列ではカメラの内部パラメータが既知である必要がある．内部パラメータを用いて，画像上の特徴点座標を正規化された同次座標で $\tilde{\mathbf{x}}_{n*} = (u_{n*}, v_{n*}, 1)$ と表現するとき，基本行列は，式 (2) である．

$$\tilde{\mathbf{x}}_{n2}^T \mathbf{E} \tilde{\mathbf{x}}_{n1} = 0, \quad \mathbf{E} = [\mathbf{t}]_\times \mathbf{R}. \tag{2}$$

基本行列は，画像間で 5 点の対応組があれば文献 [21] などの手法を用いて推定することができる．また，ソースコードも例えば [22] で公開されている．一旦基本行列 $\mathbf{E}$ が求まれば，特異値分解を用いて

$$\mathbf{E} = \mathbf{U} diag(1, 1, 0) \mathbf{V}^T \tag{3}$$

に分解でき，

$$\begin{aligned} \mathbf{R} &= \mathbf{U}\mathbf{W}\mathbf{V}^T \ \ or \ \ \mathbf{U}\mathbf{W}^T\mathbf{V}^T \\ [\mathbf{t}]_\times &= \mathbf{U}\mathbf{Z}\mathbf{U}^T \end{aligned} \tag{4}$$

として自己位置を求めることができる．ただし，

$$\mathbf{W} = \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad \mathbf{Z} = \begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \tag{5}$$

である．射影変換行列と同様に解の不定性が 4 通りあるが，前述したような条件を追加して解を絞り込む．

基本行列では，5 点の対応点組を用いて推定するが，より多くの点を用いて精度よく推定したい場合には，求めらえた回転行列と並進ベクトルから特徴点の 3 次元位置を 2.2.4 節により求め，求められた複数の点を 3.3 節のバンドル調整を用いることで精度よく求めなおすこともできる．

2.2.3 基礎行列

基礎行列は，内部パラメータが未知である場合にも基本行列と同じようにエピポーラ幾何を推定することができる．基礎行列は，基本行列ではカメラの内部パラメータから算出された正規化画像座標であったところが，画像座標または特徴点の位置関係から正規化された座標になり，以下のように表すことができる．

$$\tilde{\mathbf{x}_2}^T \mathbf{F} \tilde{\mathbf{x}_1} = 0, \quad \mathbf{F} = \mathbf{A}_2^{-T} [\mathbf{t}]_\times \mathbf{R} \mathbf{A}_1^{-1}. \tag{6}$$

基礎行列 $\mathbf{F}$ は，7 点法または 8 点法で推定することができる [23]．これらのソースコードは，OpenCV[18] に cv::findFundamentalMat の関数名で実装されている．

カメラの内部パラメータが既知であれば，基礎行列から“基本行列に近いもの”（回転行列の正規直交性などを満たさないため，厳密には同じではない．）を求め，それを基本行列と同様の手順で特異値分解してカメラの回転行列と並進ベクトルを求めることもできる．この際に，“基本行列に近いもの” から正規直交基底を持つ回転行列 $\mathbf{R}$ を特異値分解により

$$\mathbf{R} = \mathbf{U} diag(1, 1, det(\mathbf{U}\mathbf{V}^T)) \mathbf{V}^T \tag{7}$$

のように求める [24].

基礎行列は基本行列に比べて実装が容易なことと，計算時間が短いこと [25] が利点である．ただし，基本行列のように回転行列の正規直交基底を維持しつつ，求めることができないため，特徴点のノイズ等によって基礎行列から求められる回転行列と並進ベクトルの精度は基本行列のものより劣る場合もある．精度を上げたい場合には，バンドル調整 (3.3 節) の初期値に用いて，回転行列と並進ベクトルを求めなおしてもよい．

2.2.4 対応点の３次元形状

カメラの回転行列と並進ベクトルが求められれば，特徴点の組から三角測量の原理によって特徴点の３次元位置を求めることができる．これにより，マーカーがない場合であっても，カメラの自己位置と３次元形状を復元できたこととなる．ただし，射影変換，基本行列，基礎行列のいずれを用いても事前知識なしには対象やカメラの並進ベクトルの絶対的なスケールは求めることができない．大抵の場合は，並進ベクトルのノルムを 1 に正規化する等の処理を行うが，加速度センサやジャイロセンサによりスケールを決定する方法もある [26].

3. 追跡処理

前節の初期化によって，初期化に用いたカメラの自己位置と環境中の３次元形状が求められる．本節では，環境中の３次元形状が求められている状態で，移動カメラから撮影された自己位置が未知な画像が入力されたときのカメラの自己位置推定方法を説明する．

追跡処理では，最初に３次元座標がわかっている特徴点が入力画像上でどこにあるか対応付けを行う必要がある．これは，2.2 節で述べた方法を用いればよい．以降では，３次元座標が既知な特徴点と，その特徴点の対応付け結果を用いたカメラの自己位置推定について説明する．

3.1 カメラの自己位置推定

入力画像における特徴点の位置を同次座標で $\tilde{\mathbf{x}}$ と，その３次元位置を同次座標で $\tilde{\mathbf{X}}$ とするとき，透視投影行列 $\mathbf{P}$ を使って以下のように表すことができる．

$$\tilde{\mathbf{x}} = \mathbf{P}\tilde{\mathbf{X}} = \mathbf{A}\,[\mathbf{R}|\mathbf{t}]\,\tilde{\mathbf{X}} \tag{8}$$

複数の特徴点の位置とその３次元座標が既知であるとき，カメラの位置と姿勢を求める問題は PnP 問題（Perspective n-Point）として知られている [27]．この問題を求めるに

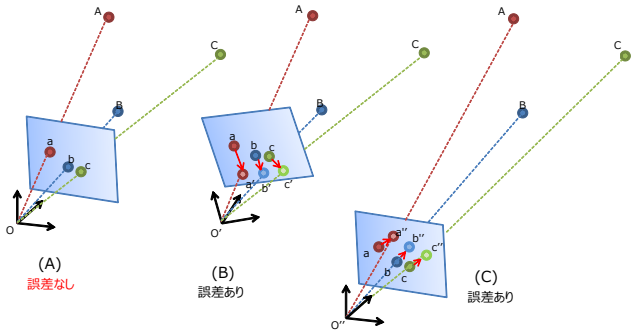

図 3 投影誤差の最小化とカメラの自己位置推定.

は，線形解法と非線形解法がある．線形解法では，少なくとも 6 つの対応点が必要である．透視投影行列は，12 個のパラメータで構成されるが，$\mathbf{P}$ の各要素を 3 行 4 列目の成分 p_{34} で除算することより，11 個のパラメータとして線形的に求めることができる [23]．これは，

$$\mathbf{Ap} = \mathbf{b} \tag{9}$$

となるように，行列 $\mathbf{A}$ と 11 次元ベクトル $\mathbf{b}$ を整理し，11 次元ベクトル $\mathbf{p} = (p_{11},,,,p_{33})$ について，最小 2 乗誤差が小さくなるように推定すればよい．

射影変換行列 $\mathbf{P}$ から回転行列 $\mathbf{R}$ と並進ベクトル $\mathbf{t}$ を求めるには，RQ 分解を行えばよい．OpenCV には，cv::decomposeProjectionMatrix の関数名で実装されている．

非線形解法では，図 3 に示すように，見つかった特徴点の位置とカメラの回転行列と並進ベクトルを用いて求められた理想的な特徴点の投影位置の差が小さくなるようにコストを設定する．もう少し具体的には，特徴点 i の 3 次元座標を同次座標表現で $\tilde{\mathbf{X}}_i$，回転行列 $\mathbf{R}$ と並進ベクトル $\mathbf{t}$ が与えられたときの画像上の投影位置 $\bar{\mathbf{x}}_i$(同次座標ではなく，画像座標) を式 (8) を用いて表現する．実際に画像上で観測された特徴点の位置を $\mathbf{x}_i$ とするとき，以下のような投影誤差を最小にする $\hat{\mathbf{R}}, \hat{\mathbf{t}}$ を求める．

$$E(\hat{\mathbf{R}}, \hat{\mathbf{t}}) = \min_{\mathbf{R},\mathbf{t}} \sum_{i \in P} |\mathbf{x}_i - \bar{\mathbf{x}}_i|^2. \tag{10}$$

ここで，P は画像に写る特徴点の集合である．3.3 節で述べる，バンドル調整では特徴点の 3 次元位置とカメラの回転行列と並進ベクトルを最適化するが，ここではカメラの運動のみを推定すればよい．解法等は，3.3 節のバンドル調整とほぼ同じため，そちらを参照されたい．

線形解法では，初期値を必要としないメリットがあるが，最小化するパラメータが実際の現象とあっていない（すなわち，非線形解法のように観測された特徴点の位置と理想的な特徴点の位置を近づけるように最小化されるわけではない）ため，推定される精度が十分でない場合がある．動画の場合には，前の時刻の位置から大きくずれていないため，回転行列と並進ベクトルの初期値は，前の時刻のものをそのまま用いればよく，線形解法は行わなくてもよい場合もある．3 次元位置と画像上の位置から，カメラの自己位置を求める処理は，OpenCV では cv::calibrateCamera，SBA[28], [29] では sba_mot_levmar の関数名で実装されている．

3.2 カメラの自己位置の再同定

滑らかに移動しながら撮影された映像を用いる場合には，一度カメラの自己位置を見失ってしまうと，処理が破綻してしまう．そのような状況を抑えるために，自己位置の再同定手法が提案されている [30], [31]．広い空間で得られた環境地図から高速かつ正確に自己位置を推定するための対応点探索手法が肝であり，一度対応点が見つかれば 3.1 節の方法により自己位置を求めることができる．

3.3 バンドル調整

連続して追跡処理を行っていくと，誤差の蓄積が起こり，カメラの自己位置や 3 次元形状に歪みが生じやすくなる．この誤差の蓄積を低減するために，バンドル調整を行う．バンドル調整は，観測された特徴点の画像上での位置と，その特徴点の 3 次元座標を用いて，カメラの位置と姿勢を推定する機能と，複数の画像に映っている同一の特徴点とその画像のカメラの運動情報から特徴点の 3 次元位置推定をする機能の 2 つがある．バンドル調整では，観測された特徴点の位置と，仮定したカメラの自己位置 $\mathbf{R}_*, \mathbf{t}_*$ と 3 次元座標 $\mathbf{X}_*$ を用いて画像上に投影した位置 $\bar{\mathbf{x}}_*$ との誤差となるように，以下のコスト関数を定義する．

$$E(\hat{\mathbf{R}}, \hat{\mathbf{t}}, \hat{\mathbf{X}}) = \min_{\mathbf{R},\mathbf{t},\mathbf{X}} \sum_{k \in C} \sum_{i \in P} |\mathbf{x}_{i,k} - \bar{\mathbf{x}}_{i,k}|^2. \tag{11}$$

ここで，

$$\tilde{\mathbf{x}}_{i,k} = \mathbf{A}[\mathbf{R}_k|\mathbf{t}_k]\tilde{\mathbf{X}}_i, \tag{12}$$

C はカメラの集合，P は特徴点の集合である．コスト関数の最小化は，$\bar{\mathbf{x}}$ を回転パラメータ (3 個) と並進パラメータ (3 個) で偏微分し，レーベンバーグマーカート法などを用いて推定する．この際に，疎行列の性質やシューア補行列を用いると計算量を削減することができる．理論的な詳細は，チュートリアル等 [32], [33] を別途参照されたい．バンドル調整のソースコードして，SBA[28], [29] や PBA[34], [35] がある．

3.4 公開ソフトウェア

単眼カメラによる SLAM のソースコードとして代表的なものに MonoSLAM[15], [36]，PTAM(Parallel Tracking and Mapping)[37], [38]，Bundler[39], [40] がある．

MonoSLAM や PTAM は，ビデオのように連続画像からのカメラの自己位置推定と 3 次元形状取得を行う．MonoSLAM は，拡張カルマンフィルタを用いて過去の画像から得られた情報を伝播させて求め，PTAM は，特徴

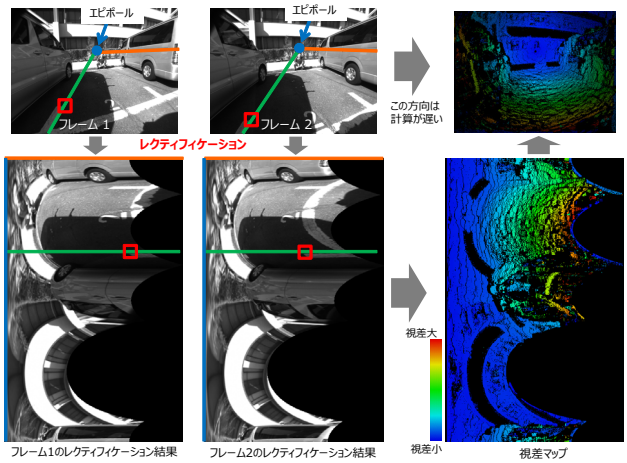

図 4 任意カメラ位置のレクティフィケーションと対応点探索結果 [3].

点の追跡結果からの姿勢推定と，バンドル調整による最適化を並列処理で行っている．

Bundler は，疎な位置関係で撮影された静止画像に対応したソフトウェアである．まばらに撮影された画像では，特徴点の見え方が大きく異なっている為，動画と比べて特徴点の追跡が難しい．そこで，見え方の変化に頑健な特徴点検出手法や特徴点対応づけとして，SIFT[17] を用いている．

4. 密な３次元形状復元

カメラの自己位置推定ができれば，エピポーラ幾何を用いることができ, 比較的少ない計算量で密な３次元形状が復元できる．密な３次元形状を復元するシンプルな方法は，２枚の画像間で構成されるエピポーラ線に沿って，ウィンドウを用いたマッチングを行い，対応点を取得する．画像内の多くの点で同じように対応点を求め，各対応点に対して三角測量の原理により，形状の復元を行う．

対応点の探索方法について，各対応点において独立に求めるローカルな対応点推定方法と，画像全体でエネルギー関数が小さくなる対応位置を求める対応点推定方法の大きく２通りある．

4.1 ローカルな対応点推定方法

ステレオカメラでは，通常は平行に近い状態でカメラが据え付けられているため，射影変換を用いたレクティフィケーションを行うことで，両画像間のエピポーラ線を平行になるように変換する [41]．このような，透視投影モデルを維持したままのレクティフィケーションでは，３次元位置の計算など従来までの手法がそのまま適用出来る．ところが，この手法はエピポールが画像中に映らず，無限に遠いところに投影されることを想定している．よって，単眼カメラが画像に対して横または縦方向に移動している場合には有効であるが，奥行き方向に移動する，すなわちエピポールがどちらかの画像に映ってしまう場合には正しく変

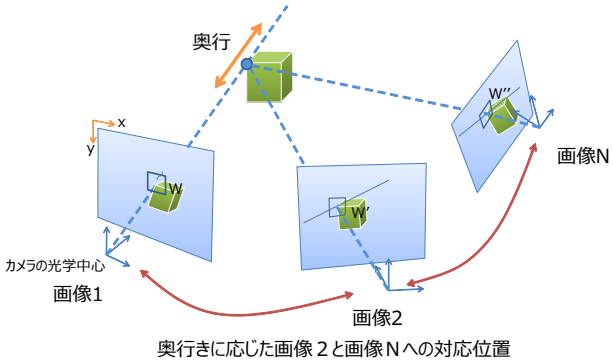

図 5 複数の画像を用いた奥行きの推定.

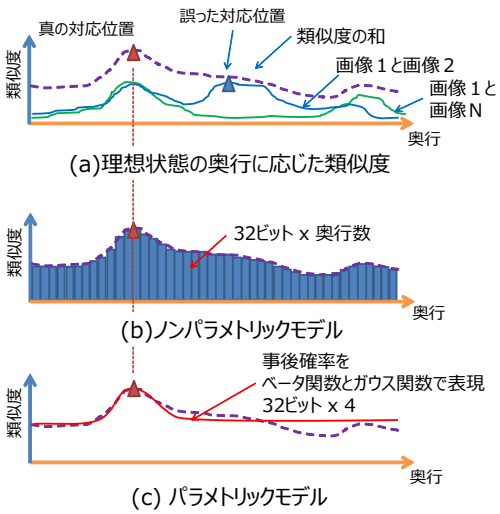

図 6 奥行きに対する類似度分布の表現方法．(a) 理想状態，(b) 類似度分布の実際のデータの持ち方，(c)Video Based MVS 法の類似度分布.

換することができない．このような任意の動きに対応したレクティフィケーション手法 [42] が提案されている．これによって，画像を平行線に合わせることができ，例えば再帰的な計算 [43] により，高速に対応点の探索を行うことができる．ただ，このレクティフィケーションでは透視投影モデルではないため，３次元位置を求めるには一度座標変換を行う必要がある．

より正確に対応点を求めるために，図 5 に示すように，多数の画像を用いる Multi-View Stereo(MVS) 法 [44] がある．この方法では，対応点を求める際の類似度を２枚にするのではなく，多数の画像に渡って加算することにより，対応点の算出ミスを減らす．

$$E(\mathbf{d}) = \sum_{k \in C} Similarity(I_1(\mathbf{x}), I_k(\mathbf{x}+\mathbf{d})) \qquad (13)$$

図 6(a) は，式 (13) を奥行きに対して連続で表現しているが，実際には奥行きは設定されたサンプリング間隔で類似度を計算する．その様子が同図 (b) であり，奥行きごとの類似度をメモリ上に保存する必要がある．例えば類似度を浮動小数点で表現すると，奥行き数 ×32 ビットのメモリが

Algorithm 1 Video Based MVS アルゴリズム

M：シード (3次元位置を求めたい場所) の最大個数
S：現在使っているシード数
for それぞれのフレーム I_k **do**
  if $S < M$ **then**
    最大 $(M - S)$ 個のシードをフレーム I_k で生成
  end if
  for それぞれのシード **do**
    (1) フレーム I_k へのシードのエピポーラ線を求める
    (2) 指定された範囲の中でウィンドウ同士の類似度を計算し，局所最大値 x_{n+1} を求める
    (3) 対応位置 x_{n+1} と $a_{n-1}, b_{n-1}, \mu_{n-1}, \sigma_{n-1}$ を用いて，$a_n, b_n, \mu_n, \sigma_n$ とインライヤの割合 $\eta_n = \frac{a_n}{a_n+b_n}$ を求める
    if $\eta_n < \eta_{outlier}$ **then**
      シードを削除する
    end if
    if $\eta_n > \eta_{outlier}$ かつ $\sigma > \sigma_n$ **then**
      3次元座標に変換して出力，シードを削除する
    end if
  end for
end for

必要である．加えて，$E(\mathbf{d})$ は，奥行きを推定する点ごとに保持する必要があるため，画像全体ではさらに多くのメモリを消費する．多くのメモリ消費は，計算できるアーキテクチャが限られることや計算機のバスの転送速度の制限等により計算時間が増大しやすい．

リアルタイム処理に向いたアルゴリズムとして，Video Based MVS(VMVS) [45], [46] がある．同手法では，類似度を図 6(c) に表すように少数のパラメータで表現する．その原理について簡単に説明する．

式 (13) を，真の奥行き Z，真の奥行き付近に測定される割合 (インライヤの割合)π，指定された奥行き $[Z_{min}, Z_{max}]$ の間で観測されるアウトライヤの割合 $1-\pi$ とする．真の奥行き Z とインライヤの割合 π が与えられたときの n 回目 (1 枚の画像が追加されるごとに式 (13) を計算していることに相当) の位置 x_n における類似度を以下のようにガウス分布 N と一様分布 U で表す．

$$p(x_n|Z,\pi) = \pi N(x_n|Z,{\tau_n}^2)+(1-\pi)U(x_n|Z_{min},Z_{max}). \quad (14)$$

このとき，観測毎に独立と仮定すると，

$$p(x_n|Z,\pi) \propto p(Z,\pi)\prod p(x_n|Z,\pi) \quad (15)$$

が成り立つ．ここで，奥行きの事後確率を以下のように近似する．

$$q(Z,\pi|a_n,b_n,\mu_n,\sigma_n) = Beta(\pi|a_n,b_n)N(Z|\mu_n,\sigma_n^2)$$
$$Beta(\pi|a,b) = \frac{\Gamma(a)\Gamma(b)}{\Gamma(a+b)}\pi^{a-1}(1-\pi)^{b-1}. \quad (16)$$

$q(Z,\pi|a_n,b_n,\mu_n,\sigma_n)$ を $n-1$ 回目の観測後の事後確率だとすると，x_n の位置に観測された後の事後確率は以下となる．

$$\propto p(x_n|Z,\pi)q(Z,\pi|a_{n-1},b_{n-1},\mu_{n-1},\sigma_{n-1}^2). \quad (17)$$

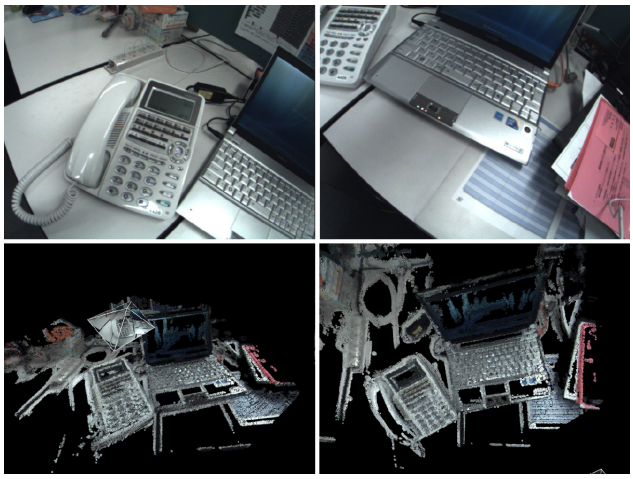

図 7 Video Based MVS 法による 3 次元復元結果 [46].

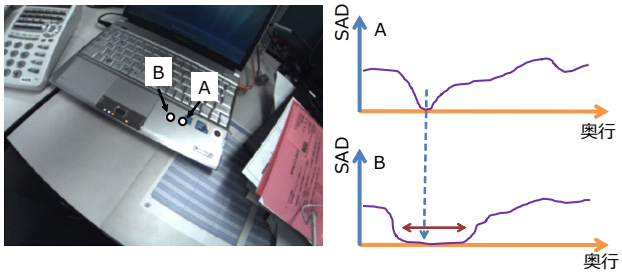

図 8 テクスチャの様子と類似度分布の形状.

n 回目の観測による事後確率をモーメントマッチング法を用いて $a_n, b_n, \mu_n, \sigma_n$ を求める．実際の計算は，$a_{n-1}, b_{n-1}, \mu_{n-1}, \sigma_{n-1}$ と x_n を用いて方程式を解くことで推定される [45]．ここまでの処理の手順をアルゴリズム 1 に示す．

これによって，従来の MVS 法では奥行きごとに保持していた類似度を保持する必要がなくなり，メモリ量を削減できる．また，従来の MVS 法では対応位置を決定するために，類似度の高い位置周辺の急峻さや，2 個目に類似度の高い位置の探索などが用いられることがあるが，VMVS 法では分散やインライヤの割合などから直接判断できるため，新たな計算が必要ない．また，類似度を計算する範囲をガウス関数の平均値周辺に限って計算したり，その周辺だけサンプリング間隔を狭くすることもできるため，高速かつ高精度に 3 次元位置を計算できる．実際に，ラップトップ PC を用いて画像の入力から毎秒 58 万点の密な 3 次元位置の推定までを 37 ミリ秒で行っている [6], [46].

図 7 の上段は，入力画像の一部を，下段は VMVS 法によって推定された 3 次元復元結果である．誤対応がほとんどなく，正しく 3 次元形状が復元されていることがわかる．また，模様のない，手前の机の領域は 3 次元形状が復元されていない．

4.2 全体最適化による対応点推定方法

前節のローカルな対応点推定方法では，特徴のある場所

は3次元形状を計算することができるが，特徴のない (テクスチャレス) 場所では対応を求めることができない．これは図8に示すように，特徴のある場所 A では類似度 (または相違度) のピークが現れて一意に対応位置を求めることができるが，特徴のない場所 B では，類似度が略一定になってしまう領域が生じ，略一定の区間で対応位置を一意に決めることができない．そこで，DTAM(Dense Tracking and Mapping)[47] ではエネルギー関数を

$$E_\xi = \int_\Omega \{g(\mathbf{x})||\nabla\xi(\mathbf{x})||_\epsilon + \lambda\mathbf{C}(\mathbf{x},\xi(\mathbf{x}))\}\,d\mathbf{x} \qquad (18)$$

としている．ここで，$g(\mathbf{x}) = e^{-\alpha||\nabla\mathbf{I}_r(\mathbf{x})||_2^\beta}$，$||\nabla\xi(\mathbf{x})||_\epsilon$ は Huber ノルム，$\mathbf{C}(\mathbf{x},\xi(\mathbf{x}))$ は画像間の類似度関数であり，同論文では SAD を採用している．このエネルギー関数を最小化することで画像中の奥行き ξ を求める．式自体は，ノイズ除去や超解像で用いるエネルギー関数とほぼ同じである．まず，式 (18) のうち，$g(\mathbf{x})||\nabla\xi(\mathbf{x})||_\epsilon$ を見ると画像の輝度が変化する (エッジが大きい) ところでは奥行きの勾配 (奥行きの変化具合)$\nabla\xi$ が大きくても，エネルギーが大きくならないことがわかる．逆に，画像の輝度の変化が少ないところで奥行きの変化が起こると，エネルギーが大きくなる．例えば，図8では，A と B の間で画像の輝度に大きな勾配の変化がないため，奥行きが変化しづらくなり，推定される B の奥行きは A の奥行き値と同じような位置になる．続いて，式 (18) の解き方を説明する．まず，補助変数 α を用いて

$$E_{\xi,\alpha} = \int_\Omega \{g(\mathbf{x})||\nabla\xi(\mathbf{x})||_\epsilon + \tfrac{1}{2\theta}(\xi(\mathbf{x})-\alpha(\mathbf{x}))^2 + \lambda\mathbf{C}(\mathbf{x},\alpha(\mathbf{x}))\}\,d\mathbf{x} \qquad (19)$$

とする．次に $g(\mathbf{x})||\nabla\xi(\mathbf{x})||_\epsilon + \mathbf{Q}(\mathbf{x})$ と $\mathbf{Q}(\mathbf{x}) + \lambda\mathbf{C}(\mathbf{x},\alpha(\mathbf{x}))$ の2つの項にする．ここで，$\mathbf{Q}(\mathbf{x}) = \frac{1}{2\theta}(\xi(\mathbf{x})-\alpha(\mathbf{x}))^2$ である．前半の項は凸関数であり，主双対問題を使って最適化を行う．後半の項は $\mathbf{x}$ について設定範囲内で全探索を行い，最適値を求める．式 (19) の最適化は，前半の項で求められたパラメータを用いて後半の項の最適値を計算する，繰り返し計算によって求める．前半，後半の各画素におけるパラメータは並列して計算することができるため，GPU の実装に向く．ただ，エネルギー最小化の過程で $C(\mathbf{x},\xi(\mathbf{x}))$ にアクセスするため，内部では各点の類似度を保持しており，VMVS 法と比べてメモリ使用量が多い．図9は，奥行きを画像全体のエネルギーが最小になるように推定した結果である．式 (18) のエネルギー最小化を3次元のボクセルに拡張した手法 [48] も提案されている．

5. 実験機材

本節では，カメラからの自己位置推定や周辺の3次元形状推定に必要な機材を選定する際の注意点について簡単に説明する．

レンズの選定　レンズは画角の広いものを用いたほうが，広い範囲から抽出した特徴点を用いることができる．これにより，処理を安定に保ちやすいことが知られている [49].

図 9　全体最適化による奥行き推定結果．(左) 入力画像の一部，(右) 奥行き推定結果.

カメラの選定　カメラは，グローバルシャッタを用いたものが望ましい．グローバルシャッタでは，撮像素子を一括して映像を取得するため，どの画素位置も同一時刻のデータである．最近の Web カメラや，スマートフォンに搭載されているカメラは，撮像素子として CMOS を用いるローリングシャッタのものが多い．ローリングシャッタでは，画像中に移動している物体があると，像が歪んでしまう．像の歪によって，画像上の特徴点の位置がずれてしまい，計測誤差の原因になるため，ローリングシャッタの影響を取り除く方法 [50] もある．

6. おわりに

本稿では，移動カメラから撮影された画像を用いた周辺環境の3次元形状復元と自己位置推定，さらに自己位置推定結果を用いて，周囲環境の緻密な3次元形状の復元手法について，一通り説明した．また，実験に用いる機材の選定の指針について簡単に説明した．

参考文献

[1] 一刈, 川野, 天目, 大島, 柴田, 田村：“映画制作を支援する複合現実型プレビジュアリゼーションとキャメラワーク・オーサリング”, 日本バーチャルリアリティ学会論文誌, **12**, 3, pp. 343–354 (2007).
[2] D. Scaramuzza and F. Fraundorfer: “Visual odometry”, IEEE Robotics & Automation Magazine, **18**, 4, pp. 80–92 (2011).
[3] A. Seki and R. Okada: “Monocular-camera based obstacle detection with measurement error estimation”, ITS World Cogress in Tokyo (2013).
[4] J. K. Suhr, H. G. Jung, K. Bae and J. Kim: “Automatic free parking space detection by using motion stereo-based 3d reconstruction”, Machine Vision and Applications, **21**, 2, pp. 163–176 (2010).
[5] 山口, 加藤, 二宮：“車載単眼カメラによる車両前方の障害物検出”, 情処学研報 (2005-CVIM-151), **2005**, 112, pp. 69–76 (2005).
[6] 関, O. Woodford, R. Gherardi, 畠山, 島村, 岡田, B. Stenger, R. Cipolla：“福島第一原子力発電所のがれき除去作業へのリアルタイム3次元再構成技術の適用”, ビジョン技術の実利用ワークショップ (ViEW) (2013).
[7] “Kinect™”, http://ja.wikipedia.org/wiki/Kinect.
[8] C. Kerl, J. Sturm and D. Cremers: “Robust odometry

estimation for RGB-D cameras", Proc. of the IEEE Int. Conf. on Robotics and Automation (ICRA) (2013).
[9] R. A. Newcombe, S. Izadi, O. Hilliges, D. Molyneaux, D. Kim, A. J. Davison, P. Kohli, J. Shotton, S. Hodges and A. Fitzgibbon: "Kinectfusion: Real-time dense surface mapping and tracking", Proc. of IEEE ISMAR (2011).
[10] P. Besl and N. McKay: "A method for registration of 3-d shapes", IEEE Trans. PAMI, **14**, 2, pp. 239–256 (1992).
[11] A. Milella and R. Siegwart: "Stereo-based ego-motion estimation using pixel tracking and iterative closest point", Proc. IEEE International Conference on Computer Vision Systems, p. 21 (2006).
[12] C. Tomasi: "Shape and motion from image streams under orthography: a factorization method", Int. J. Comput. Vision, **9**, pp. 137–154 (1992).
[13] J. Costeira and T. Kanade: "A multibody factorization method for independently moving objects", Int. J. Computer Vision, **29**, 3, pp. 159–179 (1998).
[14] H. Kato and M. Billinghurst: "Marker tracking and HMD calibration for a video-based augmented reality conferencing system", Proceedings of the 2nd International Workshop on Augmented Reality (IWAR 99), San Francisco, USA (1999).
[15] A. J. Davison, I. D. Reid, N. D. Molton and O. Stasse: "MonoSLAM: Real-time single camera SLAM", IEEE Trans. PAMI, **26**, 6, pp. 1052–1067 (2007).
[16] B. Lucas and T. Kanade: "An iterative image registration technique with an application to stereo vision", Proc.Imaging Understanding Workshop, pp. 121–130 (1981).
[17] D. Lowe: "Distinctive image features from scale-invariant keypoints", Int. J. Comput. Vision, **60**, 2, pp. 91–110 (2004).
[18] "OpenCV(Open Source Computer Vision)", http://opencv.org/.
[19] O. Faugeras and F. Lustman: "Motion and structure from motion in a piecewise planar environment", International Journal of Pattern Recognition and Artificial Intelligence, **2**, 3, pp. 485–508 (1988).
[20] 関, 奥富："ステレオ動画像を利用した平面抽出による障害物検出", 情報処理学会論文誌, **45**, SIG 13(CVIM10), pp. 1–10 (2004).
[21] D. Nistér: "An efficient solution to the five-point relative pose problem", IEEE Trans. PAMI, **26**, 6, pp. 756–777 (2004).
[22] H. Stewenius: "Calibrated fivepoint solver", http://vis.uky.edu/ stewe/FIVEPOINT/.
[23] R. Hartley and A. Zisserman: "Multiple View Geometry in Computer Vision Second Edition", Cambridge University Press (2003).
[24] K. Kanatani: "Statistical Optimization for Geometric Computation: Theory and Practice", Elsevier Science (1998).
[25] T. Botterill, S. Mills and R. D. Green: "Fast ransac hypothesis generation for essential matrix estimation", DICTA, IEEE, pp. 561–566 (2011).
[26] P. Tanskanen, K. Kolev, L. Meier, F. C. Paulsen, O. Saurer and M. Pollefeys: "Live metric 3D reconstruction on mobile phones", Proc. IEEE ICCV (2013).
[27] R. Szeliski: "Computer Vision: Algorithms and Applications", Springer (2011).
[28] M. A. Lourakis and A. Argyros: "SBA: A Software Package for Generic Sparse Bundle Adjustment", ACM Trans. Math. Software, **36**, 1, pp. 1–30 (2009).
[29] M. A. Lourakis and A. Argyros: "SBA : A generic sparse bundle adjustment C/C++ package based on the levenberg-marquardt algorithm", http://users.ics.forth.gr/ lourakis/sba/.
[30] B. Williams, G. Klein and I. Reid: "Real-time SLAM relocalisation", Proc. IEEE ICCV (2007).
[31] S. N. Sinha: "Real-time image-based 6-DOF localization in large-scale environments", Pro. IEEE CVPR, pp. 1043–1050 (2012).
[32] 岡谷, 増田, 黄瀬, 柳井, 和田, 安田, 田中："コンピュータビジョン最先端ガイド 3", アドコム・メディア株式会社 (2010).
[33] 岩元, 菅谷, 金谷："3 次元復元のためのバンドル調整の実装と評価", 情処学研報 (2011-CVIM-175), **2011**, 19, pp. 1–8 (2011).
[34] C. Wu, S. A. amd B. Curless and S. M. Seitz: "Multicore bundle adjustment", Proc. IEEE CVPR (2011).
[35] C. Wu, S. A. amd B. Curless and S. M. Seitz: "Multicore bundle adjustment", http://grail.cs.washington.edu/projects/mcba/.
[36] A. J. Davison: "MonoSLAM", http://www.doc.ic.ac.uk/ ajd/software.html.
[37] G. Klein and D. Murray: "Parallel tracking and mapping for small AR workspaces", Proc. IEEE ISMAR, Nara, Japan (2007).
[38] G. Klein: "Parallel tracking and mapping for small AR workspaces", http://www.robots.ox.ac.uk/ gk/PTAM/.
[39] N. Snavely: "Bundler:structure from motion (SfM) for unordered image collections", http://www.cs.cornell.edu/ snavely/bundler/.
[40] N. Snavely, S. M. Seitz and R. Szeliski: "Modeling the world from internet photo collections", Int. J. Comput. Vision, **80**, 2, pp. 189–210 (2008).
[41] P. Fua: "A parallel stereo algorithm that produces dense depth maps and preserves image features", Machine Vision and Applications, **6**, pp. 35–49 (1993).
[42] M. Pollefeys, R. Koch and L. V. Gool: "A simple and efficient rectification method for general motion", Proc. IEEE ICCV, pp. 496–501 (1999).
[43] 服部："車載向けステレオ画像処理技術", 東芝レビュー, **63**, 5, pp. 48–51 (2008).
[44] M. Okutomi and T. Kanade: "A multiple-baseline stereo", IEEE Trans. PAMI, **15**, 4, pp. 353–363 (1993).
[45] G. Vogiatzis and C. Hernandez: "Video-based, real-time multi-view stereo", Image and Vision Computing, Vol. 29, pp. 434–441 (2011).
[46] 関, O. Woodford, 岡田, B. Stenger, R. Cipolla："GPU を用いた単眼時系列画像からのリアルタイムで密な 3 D復元", 画像の認識・理解シンポジウム (MIRU) (2012).
[47] R. A. Newcombe, S. J. Lovegrove and A. J. Davison: "DTAM: Dense tracking and mapping in real-time", Proc. IEEE ICCV, pp. 2320–2327 (2011).
[48] A. Wendel, M. Maurer, G. Graber, T. Pock and H. Bischof: "Dense reconstruction on-the-fly.", Proc. IEEE CVPR, pp. 1450–1457 (2012).
[49] A. Davison, Y. G. Cid and N. Kita: "Real-time 3D SLAM with wide-angle vision", Proc. IFAC Symposium on Intelligent Autonomous Vehicles (2004).
[50] G. Klein and D. Murray: "Parallel tracking and mapping on a camera phone", Proc. of IEEE ISMAR, pp. 83–86 (2009).