

ステレオ画像からの高速な微小平面 3D サーフェス直接生成法

杉 本 茂 樹[†] 奥 富 正 敏[†]

本論文では、ステレオカメラを利用した直接的かつ効率的な 3 次元サーフェス生成手法を提案する。提案手法では、基準画像を三角パッチによるメッシュに分割し、メッシュ全頂点の奥行きを求めることによりサーフェス生成を行う。全頂点の奥行きを求める際には、基準画像上のパッチ内領域と、その領域が平面射影変換によって対応づけられる参照画像上の領域との SSD (Sum of Squared Differences) を考え、その SSD を全パッチについて加算した値をコスト関数として Gauss-Newton 法を用いて最適化する。さらに、そのコスト関数を ICIA (Inverse Compositional Image Alignment) の考え方を取り入れて定式化することにより、繰返し計算ごとの計算コストを低く抑えたアルゴリズムを実現する。また、メッシュを順次細かくしてゆく階層的アプローチを利用することにより、画像から安定にサーフェスを生成する。合成画像と実画像を用いた実験を通じて提案手法の有効性を示す。

A Direct and Efficient Method for Piecewise-planar Surface Reconstruction from Stereo Images

SHIGEKI SUGIMOTO[†] and MASATOSHI OKUTOMI[†]

In this paper, we propose a direct and efficient method for 3D surface reconstruction from stereo images. In the proposed method, we reconstruct a 3D surface by estimating the depths of all vertices of a mesh which is composed of piecewise triangular patches on a reference image. We minimize a cost function, which is the sum of each SSD (sum of squared differences) value between a single patch region in the reference image and the corresponding region in the other, by Gauss-Newton optimization. Additionally, we formulate the cost function by incorporating an ICIA (Inverse Compositional Image Alignment) manner for reducing the computational costs of a Hessian matrix in each iteration process. A surface is directly and robustly reconstructed from the images by a hierarchical meshing approach where the surface by a finer mesh is obtained from the initial depths previously estimated by a coarse mesh. The validity of the proposed method is demonstrated through the experiments using synthetic and real images.

1. はじめに

物体のサーフェスを取得するための代表的なアプローチは、ステレオ計測によって得られた 3 次元空間中の点群データを、ポリゴンメッシュや B-スプラインなどのサーフェスモデルにあてはめる方法である^{6),7),10)}。しかし、ステレオ計測によって得られた点群データには、一般に誤差が含まれており、このようなあてはめによって適切なサーフェスを生成することは困難とされる⁷⁾。密なステレオ計測結果を手がかりに、Oriented Particle¹²⁾ と呼ばれる微小平面を物体表面上の点にあてはめることで、サーフェスをモデル化する方法も提案されている⁴⁾。しかし、この手法で

は、先に密なステレオ計測を行った後に、誤差を含む点群データをセグメンテーションするなどの複雑な処理が必要であり、計算コストが大きい。Level Set 法や Graph Cuts 法を利用したステレオ計測手法^{2),9)} を用いて、より高精度な点群データを取得する方法も考えられるが、その計算コストはさらに大きくなる。

上述のようなステレオ計測結果に対するモデルあてはめを利用したアプローチのほかに、画像からより直接的にサーフェスを得る手法として、画像間の輝度差情報をコスト関数として表現し、そのコスト関数を最小化することでサーフェスモデルパラメータを求める方法がいくつか提案されている^{5),8),13),14)}。代表的な手法として、対象の 3 次元形状を三角ポリゴンメッシュでモデル化し、コスト関数を最小化することによって、メッシュ全頂点の 3 次元位置を求める手法⁵⁾ が知られている。しかし、この場合、解全体の自由度が大きいいため、安定な推定を行うためには、曲面の滑らかさに

[†] 東京工業大学大学院理工学研究科機械制御システム専攻
Department of Mechanical and Control Engineering,
Graduate School of Science and Engineering, Tokyo Institute of Technology

関する強い拘束を利用する必要がある。また、最適化計算の計算コストも大きい。そのほか、20 台以上のカメラを利用する手法¹⁴⁾ や、シルエット情報を利用した手法⁸⁾ などがあげられるが、これらの手法は、非常に限定された撮影環境を想定したものである。

このような既存のサーフェス生成手法は、主にフォトリアリスティックな CG 作成を目的とした技術であるため、計算コストに関しては特に考慮されていない。しかし、サーフェス生成技術は、CG 作成だけでなく、平面検出や物体認識など、さまざまなアプリケーションへの応用が可能であり、保持するデータ量の低減が必要な場合にも有効な技術である。特に、ロボットの歩行や宇宙船の着陸などでは、歩行や着陸のための地表面の平坦領域検出や、姿勢制御のための接地点の法線計測が必要不可欠といえ、サーフェス作成技術はこのような目的においても有力な手段である。サーフェス生成をさまざまなアプリケーションに応用するためには、より効率的なアルゴリズムが必要といえる。

そこで、本論文では、ステレオ画像を利用した直接的かつ効率的な 3 次元サーフェス生成方法を提案する。提案手法では、対象のサーフェスは、微小な三角パッチによって構成されるポリゴンメッシュによって近似できるものと仮定する。そして、基準画像を複数の三角パッチによる固定されたメッシュを用いて分割し、メッシュ全頂点の奥行きを求めることにより、サーフェス生成を行う。全頂点の奥行きを求める際には、基準画像上のパッチ内領域と、その領域が平面射影変換によって対応づけられる参照画像上の領域との SSD (Sum of Squared Differences) を考え、その SSD を全パッチについて加算した値をコスト関数として Gauss-Newton 法を用いて最適化する。このとき、ステレオ画像間のエピポーラ拘束の下では、パッチの平面射影変換が 3 自由度の平面パラメータで記述できることを利用し、高速平面パラメータ推定手法¹¹⁾ の考え方を拡張してコスト関数を定式化することにより、繰返し計算ごとの計算コストを低く抑えたアルゴリズムを実現する。また、提案手法では、階層的アプローチを利用し、最初は粗いメッシュを用いてサーフェスを生成し、その結果を初期値として、より細かいメッシュによる処理を順次行うことにより、初期値に依存しない安定なサーフェス生成を行う。

提案手法は、しばしば Ill-posed 問題となる密なステレオ計測に、形状の平滑化などの正則化項を導入して密なサーフェスを求める手法¹³⁾ とは異なり、微小平面サーフェスモデルを導入することにより、ステレオ計測における解の自由度を下げ、Ill-posed 問題と

なることを回避しているといえる。提案手法における解全体の自由度は、ステレオ計測に正則化を導入した場合よりもはるかに小さく、自由度に対する同様の利点を持つ文献 5) の手法と比較しても 1/3 である。同時に、提案手法では、画像上の画素をすべて利用するため、文献 5) の手法のように 3 次元空間をサンプリングする場合よりも、画素値を有効に利用でき、正則化項を利用しなくても比較的高い安定性を実現できる。さらに、メッシュの固定化と、平面射影変換を陽に利用した定式化により、固定注目領域に対する高速平面パラメータ推定手法¹¹⁾ を応用できるという利点を持つ。これにより、提案手法では、非常に多数の画素値を利用するにもかかわらず、1 秒程度でのサーフェス生成を可能にしており、従来のサーフェス生成手法が一般に数分以上の計算時間が必要であることと比較すると、画期的に高速なアルゴリズムを実現している。

以下、本論文の構成は次のとおりである。まず、2 章では、全頂点の奥行きを求めるためのコスト関数を定式化する。ここでは、三角メッシュを用いた微小平面モデルについて簡潔に述べるとともに、各パッチの平面射影変換と各頂点の奥行きとの関係を明らかにする。3 章では、2 章で述べた手法を、高速平面パラメータ推定¹¹⁾ を利用して高速化する枠組みについて説明し、奥行きを求めるための具体的なアルゴリズムについて述べる。そして、4 章において実験結果を示し、最後に本論文をまとめる。

2. 全頂点の奥行き推定に基づくサーフェス生成

本章では、提案手法のサーフェス生成の基本的な考え方を説明する。まず、三角メッシュを用いた微小平面モデルと階層的メッシュについて述べ、各パッチの平面射影変換と各頂点の奥行きとの関係を利用して、全頂点の奥行きを求めるためのコスト関数を定式化する。

2.1 三角メッシュによる表面形状モデル

提案手法では、基準画像を小さな三角形で分割してメッシュを作成する。メッシュを作成する方法は、画像全体を方形で分割した後に方形の対角線を結び三角形に再分割する方法や、画像上から特徴点を抽出した後に特徴点を母点としてドロネー分割する方法など、さまざまな方法が考えられるが、ここでは、画像領域全体が微小な三角形で分割されていれば、どのような方法を用いてもよい。正三角形によって構成されたメッシュの例を図 1 に示す。

メッシュの頂点を V_m ($m = 1, 2, \dots, M$) と表し、

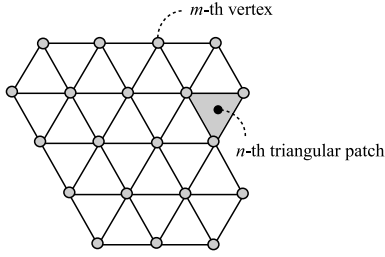


図 1 基準画像上に作成した三角メッシュの例

Fig. 1 An example of a piecewise triangular mesh on the reference image.

V_m は対象物体表面の 3 次元形状に対応した奥行き d_m を持つものとする．本論文では，全頂点の奥行き d_m ($m = 1, 2, \dots, M$) を求めることにより，対象のサーフェスを生成する．また，頂点によって構成される三角形をパッチと呼び， C_n ($n = 1, 2, \dots, N$) と表す．そして，パッチ C_n をなす 3 頂点のインデックス m の集合 $\{i, j, k\}$ を， $S(n) = \{i, j, k\}$ と書く．各パッチは 3 次元空間中において平面を構成するものとする．すなわち，パッチによって構成されるメッシュはポリゴンメッシュであり，メッシュが十分細かければ任意の 3 次元形状を精度良く近似できると考えられる．

ただし，メッシュが細かい場合は，初期値が悪いと局所解に陥りやすく，適切なサーフェスを生成することができない．そこで，本論文では，安定なサーフェス生成を行うため，メッシュを階層的に細かくしてゆくアプローチを用いる．すなわち，最初は粗いメッシュを用いてサーフェスを推定し，その結果を初期値として，より細かいメッシュを用いたサーフェス生成を順次行う．このアプローチは，最初は粗いメッシュを用いてラフなサーフェスを生成し，徐々に詳細なサーフェスを求めてゆくことを意味し，アプリケーションが必要とするサーフェスの細かさに応じて，メッシュの粗さを制御できるという利点も得られる．

2.2 サーフェス生成のためのコスト関数の定式化

基準画像上の三角パッチ C_n は，3 次元空間中で平面をなし，その 3 頂点の奥行きが定まれば，空間中の平面が定まる．このとき，基準画像上の C_n 内の画像座標 $\mathbf{u} = (u, v)^T$ は，その 3 頂点の奥行きによって定まる平面射影変換 $\mathbf{w}$ によって，参照画像上に座標変換される．よって， $\mathbf{w}$ は，画像座標 $\mathbf{u}$ と，そのパッチの 3 頂点の奥行き d_m ($m \in S(n)$) の関数として記述でき， $\mathbf{u}$ に対応する参照画像上の座標を $\mathbf{u}' = (u', v')^T$ とすると，その座標変換は， $\mathbf{u}' = \mathbf{w}(\mathbf{u}; d_{m \in S(n)})$ と書ける．

基準画像を T ，参照画像を I とすると，基準画像上のパッチ C_n のみを考えたとき，その 3 頂点の奥行きは，次式の SSD (Sum of Squared Differences) を最小にすることで求めることができる．

$$\sum_{\mathbf{u} \in C_n} \{T[\mathbf{u}] - I[\mathbf{w}(\mathbf{u}; d_{m \in S(n)})]\}^2 \quad (1)$$

全頂点の奥行きを求めるためには，式 (1) を全パッチに拡張し，次式を最小にする奥行き d_m ($m = 1, 2, \dots, M$) を求めればよい．

$$\sum_n \left[\sum_{\mathbf{u} \in C_n} \{T[\mathbf{u}] - I[\mathbf{w}(\mathbf{u}; d_{m \in S(n)})]\}^2 \right] \quad (2)$$

以下では，式 (2) 内の平面射影変換 $\mathbf{w}(\mathbf{u}; d_{m \in S(n)})$ を具体的に定式化する．ただし，ここでは，平面射影変換 $\mathbf{w}$ を直接 d_m によって記述するのではなく，まず，パッチの 3 次元空間中の平面方程式を定める 3 自由度のパラメータベクトル $\mathbf{q}_n$ を用いて表現する．これは，次章で述べるアルゴリズムの高速化のための重要なポイントとなる．その後， $\mathbf{q}_n$ を奥行き d_m によって記述する．

2.2.1 平面射影変換と平面パラメータ

以下，本論文では，煩雑さを避けるため，画像座標 $\mathbf{u}$ ， $\mathbf{u}'$ は内部パラメータが正規化された座標（正規化画像座標）を示すものとする．

いま，ある三角パッチ C_n が 3 次元空間中の平面 Π_n をなすすると， Π_n を 2 枚の画像上に射影したときの 2 枚の画像の座標関係は，同次座標表現 $\tilde{\mathbf{u}} = (u, v, 1)^T$ ， $\tilde{\mathbf{u}}' = (u', v', 1)^T$ を用いて，

$$\tilde{\mathbf{u}}' \sim \mathbf{P}_n \tilde{\mathbf{u}} \quad (3)$$

と書ける．ただし，‘ $\sim$ ’ はその両辺が定数倍の不定性を許して等しいこと（同値関係）を示し， $\mathbf{P}_n$ は 3×3 の平面射影変換行列を表している．式 (3) によって定められる座標変換を， $\mathbf{u}' = \mathbf{w}(\mathbf{u}; \mathbf{p}_n)$ と表す．ただし， $\mathbf{p}_n$ は，行列 $\mathbf{P}_n$ の要素をラスタ順に並べた 9 次元のベクトルとする．

画像座標が正規化画像座標で表されているため， $\mathbf{P}_n$ は，次式で書ける³⁾．

$$\mathbf{P}_n = \mathbf{R} + \mathbf{t} \mathbf{q}_n^T \quad (4)$$

ただし， $\mathbf{R}$ と $\mathbf{t}$ は，カメラ間の回転行列および平行移動ベクトルを表し，それぞれ既知とする．また， $\mathbf{q}_n \equiv (q_{nx}, q_{ny}, q_{nz})^T$ は，図 2 に示すように，平面 Π_n の 3 次元空間中の方程式を表す 3 自由度のパラメータ（以下，平面パラメータと呼ぶ）である．式 (4) は，射影変換パラメータ $\mathbf{p}_n$ が $\mathbf{q}_n$ の関数となることを表しており，座標変換は， $\mathbf{u}' = \mathbf{w}(\mathbf{u}; \mathbf{p}_n(\mathbf{q}_n))$ と書ける．

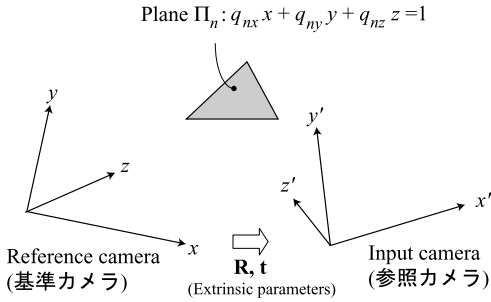


図 2 ステレオカメラ配置と平面パラメータ

Fig. 2 Stereo configuration and plane parameters.

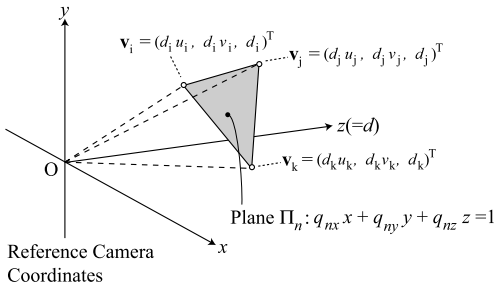


図 3 パッチ頂点の 3 次元座標

Fig. 3 3D positions of patch vertices.

2.2.2 平面パラメータと 3 頂点の奥行き

パッチ C_n の 3 頂点の基準画像上の座標を $\mathbf{u}_m = (u_m, v_m)^T$ ($m \in S(n) = \{i, j, k\}$) とし、それらの奥行きを d_m とすると、図 3 に示すように、3 頂点の 3 次元空間中の座標 $\mathbf{v}_m$ は $\mathbf{v}_m = (d_m u_m, d_m v_m, d_m)^T$ ($m \in \{i, j, k\}$) と表すことができる。このとき、 $\mathbf{q}_n$ と d_m ($m \in \{i, j, k\}$) の関係は、次式で表される。

$$\mathbf{q}_n = \mathbf{L}_n \boldsymbol{\gamma}_n \quad (5)$$

$$\text{where } \mathbf{L}_n \equiv \begin{bmatrix} u_i & v_i & 1 \\ u_j & v_j & 1 \\ u_k & v_k & 1 \end{bmatrix}^{-1}, \quad (6)$$

$$\boldsymbol{\gamma}_n \equiv (1/d_i, 1/d_j, 1/d_k)^T. \quad (7)$$

式 (5) は、 $\mathbf{q}_n$ が、 $\boldsymbol{\gamma}_n = (1/d_i, 1/d_j, 1/d_k)^T$ の関数として表現できることを意味しており、座標変換は、 $\mathbf{u}' = \mathbf{w}(\mathbf{u}; \mathbf{p}_n(\mathbf{q}_n(\boldsymbol{\gamma}_n)))$ と書ける。

2.2.3 全頂点の奥行き推定

式 (5) は、平面パラメータ $\mathbf{q}_n$ が、3 頂点の奥行きの逆数を要素としたパラメータベクトルに比例することを意味している。よって、式 (2) のコスト関数を、全頂点の奥行きの逆数を要素とするベクトルを推定するように、改めて書き換える。パラメータベクトルを、 $\boldsymbol{\Gamma} \equiv (1/d_1, 1/d_2, \dots, 1/d_M)^T$ とすると、式 (2) は次式のように書ける。

$$\sum_n \left[\sum_{\mathbf{u} \in C_n} \{T[\mathbf{u}] - I[\mathbf{w}(\mathbf{u}, \mathbf{p}_n(\mathbf{q}_n(\boldsymbol{\Gamma})))]\}^2 \right] \quad (8)$$

式 (8) は、 $\boldsymbol{\Gamma}$ をパラメータとしたコスト関数であるため、最適化手法を用いて最小化することにより、全パッチの奥行きを求めることができる。

2.3 最適化における計算コスト

式 (8) は、全頂点の奥行きに関する簡潔な式であり、非常にシンプルなアイデアによって導かれたものである。しかし、式 (8) を最小化する奥行きを求めるには、膨大な計算コストを必要とする。以下では、最適化手法として Gauss-Newton 法を利用する場合について説明する。

奥行きパラメータ $\boldsymbol{\Gamma}$ を、現在値 (初期値) $\boldsymbol{\Gamma}_0$ と微小変化量 $\boldsymbol{\Gamma}_\Delta$ を用いて、 $\boldsymbol{\Gamma} = \boldsymbol{\Gamma}_0 + \boldsymbol{\Gamma}_\Delta$ とする。Gauss-Newton 法に基づき、 $T[\mathbf{u}] - I[\mathbf{w}(\mathbf{u}, \mathbf{p}_n(\mathbf{q}_n(\boldsymbol{\Gamma})))]$ を 1 次テイラー展開し、式 (8) を $\boldsymbol{\Gamma}_\Delta$ で微分してゼロとくと、最終的に次式が得られる。

$$\boldsymbol{\Gamma}_\Delta = -\mathbf{H}^{-1} \mathbf{b}, \quad (9)$$

where

$$\mathbf{H} \equiv - \sum_n \left\{ \sum_{\mathbf{u} \in C_n} [(g\mathbf{J}\mathbf{K}\bar{\mathbf{L}}_n)^T (g\mathbf{J}\mathbf{K}\bar{\mathbf{L}}_n)] \right\}, \quad (10)$$

$$\mathbf{b} \equiv \sum_n \left\{ \sum_{\mathbf{u} \in C_n} [(g\mathbf{J}\mathbf{K}\bar{\mathbf{L}}_n)^T e] \right\} \quad (11)$$

ただし、 e は、2 枚の画像間の画素値の差であり、 $e = T[\mathbf{u}] - I[\mathbf{w}(\mathbf{u}; \mathbf{p}(\mathbf{q}(\boldsymbol{\Gamma}_0)))]$ で表される。また、 g 、 $\mathbf{J}$ 、 $\mathbf{K}$ は、それぞれ次式で表される。

$$\mathbf{g} \equiv \left[\frac{\partial I}{\partial \mathbf{w}} \right]_{\boldsymbol{\Gamma}=\boldsymbol{\Gamma}_0}, \quad \mathbf{J} \equiv \left[\frac{\partial \mathbf{w}}{\partial \mathbf{p}} \right]_{\boldsymbol{\Gamma}=\boldsymbol{\Gamma}_0}, \quad (12)$$

$$\mathbf{K} \equiv \left[\frac{\partial \mathbf{p}}{\partial \mathbf{q}} \right]_{\boldsymbol{\Gamma}=\boldsymbol{\Gamma}_0} \quad (13)$$

$\mathbf{g}$ は参照画像 I の勾配を示す 1×2 の列ベクトル、 $\mathbf{J}$ は、画像座標を射影変換パラメータ $\mathbf{p}$ で微分した結果を示す 2×9 の行列である。これらは、どちらも画素ごとに異なる値を持ち、繰返し計算のたびに再計算が必要となる。また、 $\mathbf{K}$ は、式 (4) の関係から得られる 9×3 の行列である。これは、一度計算したら再計算は不要である。 $\bar{\mathbf{L}}_n$ は、式 (6) の列数を M (頂点数) に拡張した $3 \times M$ の行列であり、 n 番目のパッチに対応する m ($m \in S(n)$) の列以外はゼロになる。すなわち、 $m \in S(n) = \{i, j, k\}$, $(1 \leq i < j < k \leq M)$ とすると、

$$\bar{\mathbf{L}}_n = \begin{bmatrix} l_{1i}^{(n)} & l_{1j}^{(n)} & l_{1k}^{(n)} \\ 0_{1,i} l_{2i}^{(n)} & 0_{i+1,j} l_{2j}^{(n)} & 0_{j+1,k} l_{2k}^{(n)} & 0_{k,M} \\ l_{3i}^{(n)} & l_{3j}^{(n)} & l_{3k}^{(n)} \end{bmatrix}, \quad (14)$$

$$\text{where } \mathbf{L}_n = \begin{bmatrix} l_{1i}^{(n)} & l_{1j}^{(n)} & l_{1k}^{(n)} \\ l_{2i}^{(n)} & l_{2j}^{(n)} & l_{2k}^{(n)} \\ l_{3i}^{(n)} & l_{3j}^{(n)} & l_{3k}^{(n)} \end{bmatrix} \quad (15)$$

となる。ただし、 $0_{a,b}$ は $3 \times (b-a)$ のゼロ行列を示す。行列 $\bar{\mathbf{L}}_n$ は、パッチごと値が異なるものの、頂点の画像座標のみに依存しているため、再計算は不要となる。

注意すべき点は、この手法では、画素ごとに異なる g と J の値を繰返し計算ごとに求める必要があるため、そのつどヘッセ行列 $\mathbf{H}$ を式 (10) のように計算する必要があり、計算コストが非常に大きいことである。これに対し、次章では、精度を損なうことなく $\mathbf{H}$ を簡単に計算するため、単一平面に対する高速平面パラメータ推定法¹¹⁾ の考え方を、複数の三角パッチに適用するように拡張する。

3. 高速化

式 (3) によって表現される平面射影変換は、行列の現在値 $\mathbf{P}_0$ と微小変化量 $\mathbf{P}_\Delta$ を用いて、次式のように表すことができる¹⁾。

$$\tilde{\mathbf{u}}' \sim \mathbf{P}_0(\mathbf{I} + \mathbf{P}_\Delta)^{-1} \tilde{\mathbf{u}} \quad (16)$$

このとき、画像の座標変換 $\mathbf{w}$ は、 $\mathbf{P}_0$ による座標変換 $\mathbf{w}(\mathbf{u}, \mathbf{p}_0)$ と、 $(\mathbf{I} + \mathbf{P}_\Delta)$ による座標変換 $\mathbf{w}_\Delta(\mathbf{u}, \mathbf{p}_\Delta)$ を用いて、 $\mathbf{w}(\mathbf{u}, \mathbf{p}) = \mathbf{w}(\mathbf{u}, \mathbf{p}_0) \circ \mathbf{w}_\Delta(\mathbf{u}, \mathbf{p}_\Delta)^{-1}$ と表すことができる。ただし、 $\mathbf{p}_0$ と $\mathbf{p}_\Delta$ は、それぞれ $\mathbf{P}_0$ と $\mathbf{P}_\Delta$ の要素をラスタ順に並べたベクトルである。画像間の平面射影変換パラメータを求める ICIA (Inverse Compositional Image Alignment) アルゴリズムでは、平面射影変換のこの特徴を利用して、繰返し計算ごとに微分要素の再計算不要なアルゴリズムを実現している¹⁾。

また、平面パラメータ $\mathbf{q}$ を求める場合には、現在値 $\mathbf{q}_0$ と初期値 $\mathbf{q}_\Delta$ を用いて $\mathbf{q} = \mathbf{q}_0 + \mathbf{q}_\Delta$ とし、 $\mathbf{p}_\Delta$ を $\mathbf{q}_\Delta$ の関数として表すと、ICIA の場合と同様に、高速なアルゴリズムを実現できることが示されている¹¹⁾。

以下では、文献 11) の方法を拡張し、射影変換パラメータの微小変化量 $\mathbf{p}_\Delta$ を、それぞれ、メッシュ全頂点の奥行きパラメータの微小変化量 Γ_Δ を用いて表すことにより、高速化効果が得られることを示す。

まず、式 (8) における $\mathbf{w}(\mathbf{u}, \mathbf{p}_n(\mathbf{q}_n(\Gamma)))$ を、

$$\mathbf{w}(\mathbf{u}, \mathbf{p}_n(\mathbf{q}_n(\Gamma))) = \mathbf{w}(\mathbf{u}, \mathbf{p}_{n0}(\mathbf{q}_{n0}(\Gamma_0))) \circ \mathbf{w}_\Delta(\mathbf{u}, \mathbf{p}_{n\Delta}(\mathbf{q}_{n\Delta}(\Gamma_\Delta)))^{-1} \quad (17)$$

のように合成変換として表現することを考える。このとき、 $\mathbf{p}_{n\Delta}$ と $\mathbf{q}_{n\Delta}$ の関係は、次式で書ける¹¹⁾。

$$\mathbf{P}_{n\Delta} = -\frac{1}{1 + \mathbf{q}_{n0}^T \mathbf{R}^T \mathbf{t} + \mathbf{q}_{n\Delta}^T \mathbf{R}^T \mathbf{t}} \mathbf{R}^T \mathbf{t} \mathbf{q}_{n\Delta}^T \quad (18)$$

また、 $\mathbf{q}_{n\Delta}$ と $\Gamma_{n\Delta}$ の関係は、式 (5) により、線形関係であることが容易に得られる。これにより、式 (8) は、次式に書き換えることができる。

$$\sum_n \sum_{\mathbf{u} \in C_n} (T[\mathbf{w}_\Delta(\mathbf{u}, \mathbf{p}_{n\Delta}(\mathbf{q}_{n\Delta}(\Gamma_\Delta))]) - I[\mathbf{w}(\mathbf{u}, \mathbf{p}_{n0}(\mathbf{q}_{n0}(\Gamma_0))])^2 \quad (19)$$

$T[\mathbf{w}_\Delta(\mathbf{u}; \mathbf{p}_{n\Delta}(\mathbf{q}_{n\Delta}(\Gamma_\Delta)))]$ をテイラー展開し、式 (19) を Γ_Δ で微分してゼロとおくことにより、式 (19) を最小にする Γ_Δ は次式で得られる。

$$\Gamma_\Delta = -\mathbf{H}^{-1} \mathbf{b}, \quad (20)$$

where

$$\mathbf{H} \equiv -\sum_n \left\{ \frac{1}{\kappa_n^2} \sum_{\mathbf{u} \in C_n} [(g\mathbf{JK}\bar{\mathbf{L}}_n)^T (g\mathbf{JK}\bar{\mathbf{L}}_n)] \right\}, \quad (21)$$

$$\mathbf{b} \equiv \sum_n \left\{ \frac{1}{\kappa_n} \sum_{\mathbf{u} \in C_n} [(g\mathbf{JK}\bar{\mathbf{L}}_n)^T e] \right\} \quad (22)$$

ここで、 e は、2 枚の画像間の画素値の差であり、式 (11) と同じものである。ただし、 g 、 J および K は、式 (11) とは異なり、以下のように書ける。

$$g \equiv \left[\frac{\partial T}{\partial \mathbf{w}_\Delta} \right]_{\Gamma_\Delta=0}, \quad (23)$$

$$J \equiv \left[\frac{\partial \mathbf{w}_\Delta}{\partial \mathbf{p}_\Delta} \right]_{\Gamma_\Delta=0} = \begin{bmatrix} u & v & 1 & 0 & 0 & 0 & -u^2 & -uv & -u \\ 0 & 0 & 0 & u & v & 1 & -uv & -v^2 & -v \end{bmatrix}, \quad (24)$$

$$K \equiv \kappa_n \left[\frac{\partial \mathbf{p}_\Delta}{\partial \mathbf{q}_\Delta} \right]_{\Gamma_\Delta=0} = \begin{bmatrix} \alpha_1 & 0 & 0 & \alpha_2 & 0 & 0 & \alpha_3 & 0 & 0 \\ 0 & \alpha_1 & 0 & 0 & \alpha_2 & 0 & 0 & \alpha_3 & 0 \\ 0 & 0 & \alpha_1 & 0 & 0 & \alpha_2 & 0 & 0 & \alpha_3 \end{bmatrix}^T, \quad (25)$$

where

$$\alpha_1 \equiv t_x r_1 + t_y r_4 + t_z r_7, \quad (26)$$

$$\alpha_2 \equiv t_x r_2 + t_y r_5 + t_z r_8, \quad (27)$$

$$\alpha_3 \equiv t_x r_3 + t_y r_6 + t_z r_9. \quad (28)$$

ただし、 $\{r_1, r_2, \dots, r_9\}$ は、 $\mathbf{R}$ の要素をラスタ順に並べたものであり、 $(t_x, t_y, t_z)^T \equiv \mathbf{t}$ である。また、 κ_n

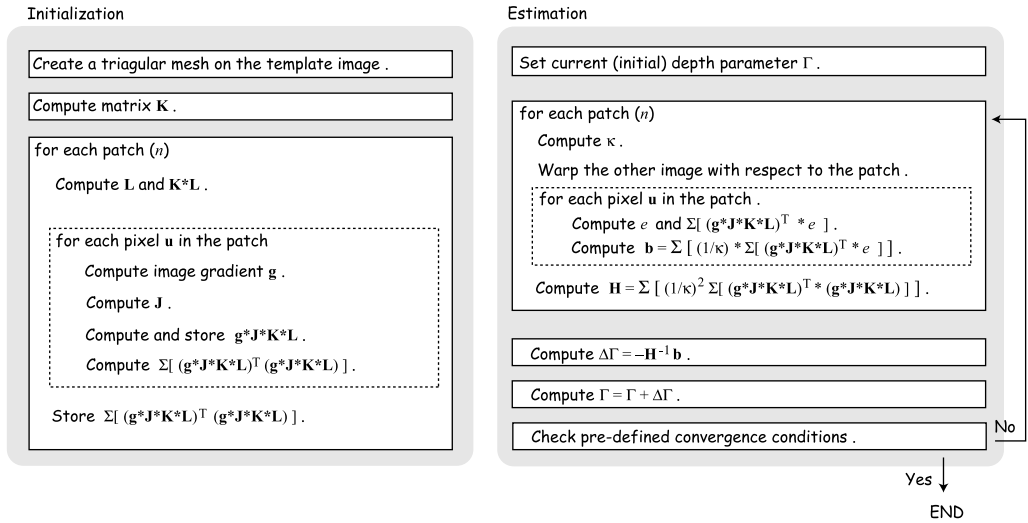


図 4 アルゴリズムフロー

Fig. 4 Algorithm flow.

は次式で書ける¹¹⁾。

$$\kappa_n \equiv -(1 + \mathbf{q}_{n0}^T \mathbf{R}^T \mathbf{t}), \quad (29)$$

式 (8) を用いた場合と異なる主な点は、 $\mathbf{g}$ が基準画像の勾配を表し、 $\mathbf{J}$ も画素値のみに依存しているため、どちらも一度計算すれば再計算が不要な点となる。ただし、ICIA による平面射影変換行列の推定¹⁾や、平面パラメータ推定¹¹⁾の場合とは異なり、ここでは、ヘッセ行列 $\mathbf{H}$ を固定化することができない。これは、 κ_n が平面パラメータの現在値 $\mathbf{q}_{n0}^T$ に依存（すなわち Γ_0 に依存）しているため、繰返し計算ごとに、係数 κ_n を再計算する必要が生じるからである。

しかし、 $\mathbf{H}$ の計算では、行列 $\sum[(\mathbf{g}\mathbf{J}\mathbf{K}\bar{\mathbf{L}}_n)^T (\mathbf{g}\mathbf{J}\mathbf{K}\bar{\mathbf{L}}_n)]$ は各パッチごとに異なるものの、繰返しごとに変化せず、さらに、この行列は実質的に $3 \times 3 = 9$ 個の要素しか持っていないため、全パッチに関して加算しても計算量は比較的小さい。また、 $\mathbf{b}$ の計算は、実質的に画素数に依存しており、単一平面のパラメータ推定や、単一平面の 3 頂点の奥行き推定を、同じ画素数（同じ画像領域サイズ）を用いて計算した場合とほぼ同じ計算量となる。すなわち、式 (20) の計算量全体を評価すると、同じ画素数を用いて単一平面のパラメータ推定をした場合と比較して、 $\mathbf{H}$ の逆行列計算の分だけが計算コストの増加といえる。ただし、 $\mathbf{H}$ の行列サイズは、 $M \times M$ であり、頂点の数が多くなるほど、その計算量は大きくなる。

本章の最後に、提案する微小平面モデルを用いたサーフェス生成アルゴリズムを図 4 にまとめる。このアルゴリズムは、初期化処理と推定処理に分けられる。初期化処理では、推定処理で利用する固定値（繰

返しごとに変化しない行列やベクトル）を計算している。また、推定処理では、式 (20) の計算に必要な値を各パッチについて求め、 Γ_Δ の計算と Γ の更新を収束するまで繰り返す。

4. 実験結果

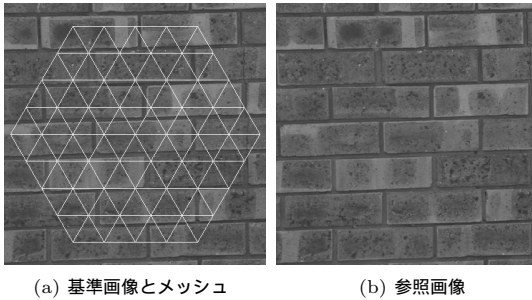
本章では、提案手法の有効性を確認するため行った合成画像を利用したシミュレーション実験と、実画像実験の結果を示す。

4.1 シミュレーション実験

シミュレーション実験では、固定メッシュを利用した場合の様子と、その推定精度について検証し、その後、階層的メッシュを利用したサーフェス生成の結果を示す。また、提案手法を高速化した場合と高速化しない場合の計算時間を示す。

4.1.1 固定メッシュによるサーフェス生成

図 5 は、基準画像の光軸上 15 m 先を中心とした半径約 7 m の球体に、テクスチャを貼り付けることによって生成したステレオ画像である（実形状は図 7 (f) を参照）。同図 (a), (b) は左画像（基準画像）と右画像を示している。同図 (a) に示すように、1 辺 50 [pixel] のメッシュを基準画像上に生成し、提案手法による表面モデル生成を行った。このときメッシュの頂点数は 61、パッチ数は 96 であった。また、頂点の奥行き初期値は、全頂点において 10 m とした。また、ステレオカメラの内部パラメータは、左右のカメラとも同じであり、次式の $\mathbf{A}$ で示す。



(a) Reference image and mesh (b) The other image

図 5 合成画像 1: シミュレーションに利用した 420×420 サイズの基準画像と参照画像。基準画像の光軸上 15 m 先を中心とした半径約 7 m の球体に、テクスチャを貼り付けることによって生成。(a) 上のメッシュは 1 辺 50 ピクセルの正三角形を用いて生成したもの ($M = 61, N = 96$)

Fig. 5 Synthetic stereo images with 420×420 pixel size: (a) is the reference and (b) is the other. The target object is a texture-mapped sphere with seven meter radius and its center at $z = 15$ meters in the reference camera coordinate system. The mesh in the reference image comprises regular triangles with 50 pixels long sides ($M = 61, N = 96$).

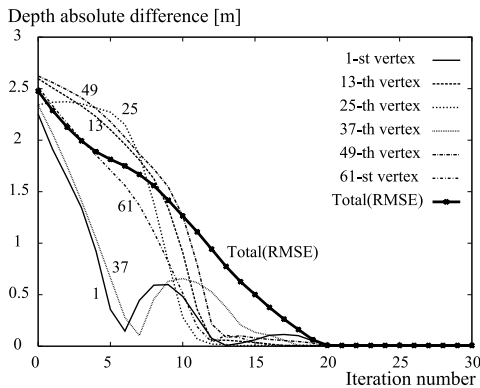


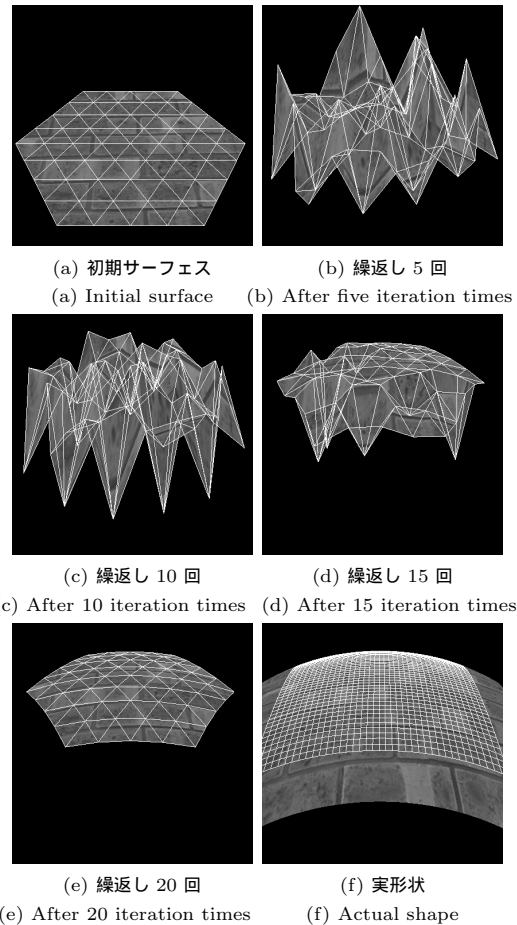
図 6 繰返し計算による頂点の奥行き誤差の変化: 6 個の頂点の奥行き誤差の絶対値 (6 本の細線) および全頂点の RMSE (太線) を表示

Fig. 6 Depth error variation by iteration: The absolute errors of six vertex depths and the RMSE of all 61 depths are shown by the six thin lines and the thick line, respectively.

$$\mathbf{A} = \begin{bmatrix} 600 & 0 & 219.5 \\ 0 & 600 & 219.5 \\ 0 & 0 & 1 \end{bmatrix} \quad (30)$$

また、カメラの外部パラメータは、 $\mathbf{R} = \mathbf{I}, \mathbf{t} = (0.3, 0, 0)^T$ とした。

図 6 は、繰返し計算による頂点の奥行き誤差 (真値との誤差の絶対値) の変化を示している。同図は、61 個の頂点のうち、ある 6 点の奥行きの誤差変化とともに、全頂点に関する RMSE (Root Mean Squared



(e) 繰返し 20 回 (f) 実形状
(e) After 20 iteration times (f) Actual shape

図 7 繰返し計算によるサーフェス変化の様子

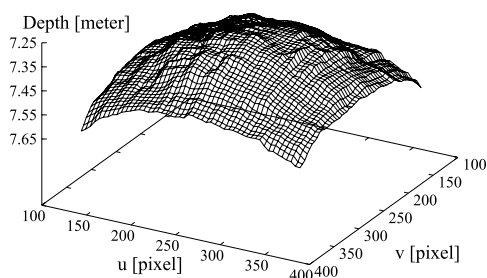
Fig. 7 Surface variation by iteration.

Error) を太線で表している。同図により、約 20 回の繰返し計算の後に、推定が適切に行われていることが分かる。

また、図 7 は、繰返し計算によるサーフェスの変化の様子を CG で示したものである。同図 (a) は初期形状 (全頂点の奥行き 10 [m]) を示し、(b) ~ (e) は、それぞれ 5, 10, 15, 20 回の繰返し計算後に推定されたサーフェスを示している。また、(f) は真の形状を示す。初期状態では、各頂点の奥行きは、実形状のものと大きく離れているが、提案手法により、最終的に真の形状をよく表したサーフェスが生成されることが分かる。

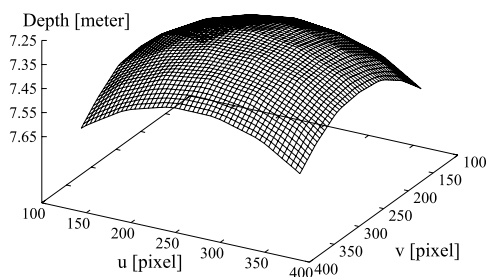
4.1.2 奥行きの推定精度

図 8 (a), (b) は、平行カメラによる合成画像 (図 5) に、標準偏差 2 (gray level) の画像ノイズを加え、ステレオ計測と提案手法による推定を行った結果をそれぞれ示している。同図では、ステレオ計測によって得ら



(a) ステレオ計測 (ウィンドウサイズ 33×33)

(a) Stereo matching (33×33 pixel window)



(b) 提案手法 (50pixel のメッシュ)

(b) Proposed method (mesh of triangles with 50 pixel long sides)

図 8 ステレオ計測と提案手法による形状推定結果の比較: 図 5 の画像に標準偏差 2[gray level] のノイズを加え, (a) は, 33×33 のウィンドウを用いた SSD によるステレオ計測の結果, (b) は, 1 辺 50 pixel のメッシュを用いた提案手法の結果を示す. 5×5 画素ごとの奥行きをつないだメッシュとして表示

Fig. 8 Comparison of stereo matching and proposed method when image noise level $\sigma = 2$ [gray level]: (a) Result of SSD-based stereo matching with 33×33 pixel window. (b) Result of proposed method by mesh of triangles with 50 pixel long sides. We plotted every 5×5 pixel depth as a rectangular mesh.

れた奥行きと, 提案手法によって得られたサーフェス上の点の奥行きを, それぞれ基準画像中央の 250×250 ピクセルの領域について 5×5 画素ごとの点を抽出し, 隣接画素の値をつなげたメッシュとして表示している. ステレオ計測では, SSD を類似度評価とし, 33×33 のウィンドウによるマッチングを行い, パラボラフィッティングによるサブピクセル推定を行った. また, 提案手法では, 前節と同様に, 1 辺 50 ピクセルのパッチによるメッシュを利用した. このとき, ステレオ計測におけるウィンドウ内の画素数は, 提案手法における 1 つのパッチ内の画素数とほぼ同数となる.

ステレオ計測の場合, 画像変形が平行移動に限定されることや, 画像ノイズなどの影響を受け, 奥行き推定に誤差が発生し, 推定された形状には細かい凹凸が見られる. これに対し, 提案手法では, 微小領域ごと

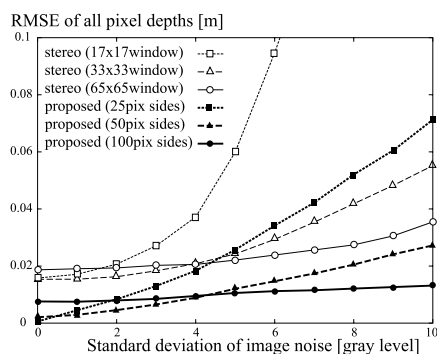


図 9 画像ノイズレベルに対するステレオ計測と提案手法の奥行き推定精度: 提案手法で与えるメッシュがサポートする全画素について, 真値との RMSE をプロット

Fig. 9 Comparison of depth error with respect to image noise level between stereo matching and proposed method: We computed the RMSE between the actual depths and the estimated depths over the pixels supported by the mesh used in the proposed method.

の画像間の平面射影変換を表現でき, メッシュ頂点を共有するという拘束が働いているため, 滑らかな面を推定できていることが分かる.

図 9 は, 同様の実験におけるステレオ計測および提案手法の奥行き推定精度を示したものである. 画像に与えたノイズの標準偏差を縦軸に示し, 提案手法で与えたメッシュがサポートする全画素について, ステレオ計測または提案手法によって得られた奥行きと真の奥行きとの RMSE (Root Mean Squared Error) を縦軸に示している. ここでは, 各ノイズ標準偏差についてステレオ画像を 100 ペア作成して RMSE を平均した. ステレオ計測では, 17×17, 33×33, 65×65 のウィンドウサイズを利用した場合を評価し, 提案手法では, 三角パッチの 1 辺が 25, 50, 100 ピクセルの場合について, 全頂点に 10 [m] の奥行きを初期値として与え, 30 回の繰返しを行った. このとき, ステレオ計測におけるウィンドウの画素数は, 提案手法における 1 つのパッチの画素数とそれぞれ近い値となる.

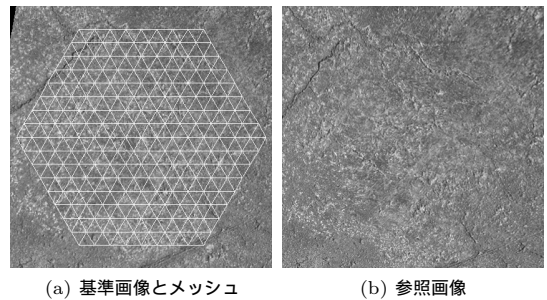
ステレオ計測と提案手法のどちらの場合も, 小さなウィンドウ (またはパッチサイズ) を用いた場合は, 画像のノイズレベルが小さい場合には精度が良く, ノイズレベルが大きくなると推定精度が著しく悪くなる傾向がある. また, どちらの手法においても, 大きなウィンドウ (またはパッチサイズ) を用いた場合は, 小さいウィンドウと比較して, 画像ノイズが小さいときの推定精度は低いもののノイズの影響をあまり受けず, ほぼ一定精度の推定ができることが分かる. しかし, ステレオ計測では, 隣接画素間の奥行きに拘束が

ないため、たとえノイズレベルがゼロであっても、表面に細かい凹凸が発生する。これに対し、提案手法では、平滑化拘束が働いたため凹凸を低減することができ、同一のウィンドウ内画素数のステレオ計測と比較した場合、より高精度な推定が可能であることが分かる。

この実験により、提案手法では、ノイズレベルがゼロのとき、小さなパッチサイズを用いることにより、非常に高精度な推定ができていることが分かる。ただし、小さいパッチサイズを用いた場合は、ステレオ計測と同様に、画像ノイズに敏感に反応し、推定精度が著しく悪くなることも示している。提案手法では、個々のパッチがローカルな領域にマッチするように変形しようとし、すべてのパッチが適切にマッチした場合に望ましいサーフェスが得られる。しかし、画像ノイズが大きい場合や、初期状態が真の形状と大きく離れていた場合、個々のパッチが正しい変形量を見つけれず、特定の頂点を（エッジ方向に）押し合ったり引き合ったりしてしまい、結果として局所解に陥り正しいサーフェスを得られないことがある。小さなパッチを利用した場合は、解の自由度が大きいため、この傾向はより高くなる。一方で、上述の実験により、大きなパッチを利用することにより、局所解への陥りを抑制できることが分かる。このことから、初期値が問題となっている場合は、はじめに大きなパッチを利用し、その結果をより小さいメッシュの初期値として利用する方法が有効であると予想できる。次項では、このような、階層的メッシュによるサーフェス生成が有効であることを実験的に示す。

4.1.3 階層的メッシュによるサーフェス生成

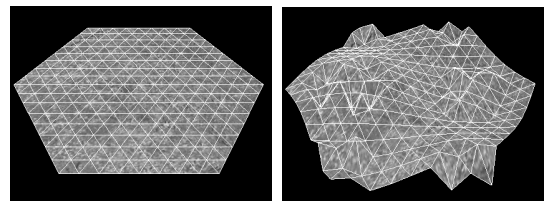
図 10 (a), (b) は、三角関数を利用して作成した 3 次元形状にテクスチャを貼り付けることによって生成したステレオ画像である（実形状は図 12 (f) を参照）。このような単調なテクスチャを利用した場合、同じ模様のテクスチャがさまざまな場所に存在するために、最初から細かいメッシュを利用すると、サーフェス生成に失敗することがある。本実験では、1 辺 200 [pixel] のパッチによって生成された頂点数 7、パッチ数 6 のメッシュを用いて、全頂点の奥行きを 10 [m] に設定して推定を行い、その結果を初期値として、パッチの 1 辺を半分にしたメッシュによる推定を順次繰り返した。最後のサーフェス生成に用いたメッシュ（1 辺 25 [pixel] のパッチによって生成された頂点数 217、パッチ数 384 のもの）を同図 (a) に示す。また、本実



(a) 基準画像とメッシュ (b) 参照画像

図 10 合成画像 2：階層的メッシュによるサーフェス生成のシミュレーションに利用した 420×420 サイズの基準画像と参照画像。三角関数によって生成した形状にテクスチャを貼り付けることによって生成。(a) 上のメッシュは、最も細かいメッシュを示し、1 辺 25 ピクセルの正三角形を用いて生成したもの ($M = 217, N = 384$)。

Fig. 10 Synthetic stereo images with 420×420 pixel size: (a) and (b) are the reference image and the other, respectively, used for our hierarchical meshing approach. The target is a texture-mapped shape created by trigonometric functions. The mesh in the reference image represents the finest mesh that comprises regular triangles with 25 pixels long sides ($M = 217, N = 384$).



(a) 初期サーフェス (b) 推定結果
(a) Initial surface (b) Result

図 11 階層的メッシュを利用せずに推定を行った場合のサーフェス生成の結果：繰返し計算の終了条件は $|\Delta\Gamma| < 10^{-4}$ として、181 回行った後の結果 ($M = 217, N = 384$)

Fig. 11 Surface estimated without hierarchical meshing: Result after 181 iteration times. We stopped the iteration process when $|\Delta\Gamma| < 10^{-4}$. ($M = 217, N = 384$)

験では、 $|\Delta\Gamma|$ の値が 10^{-4} 以下になった場合に繰返し計算を終了するという収束条件を利用した。

図 11 は、階層的メッシュを利用せず、最も細かいメッシュのみを適用した実験結果である。同図 (a) は初期状態を示し、(b) は得られたサーフェスを示している。この場合、上述の収束条件では 181 回もの繰返しを行っている。このような多数の繰返しを行ったにもかかわらず、局所解に陥り滑らかなサーフェスを生成できていない。

これに対し、図 12 は、階層的メッシュによって生成されたサーフェスを示している。同図 (a) は初期形

画像にノイズを与えなくても、合成画像生成の際のテクスチャ補間によって生じるノイズが存在している。

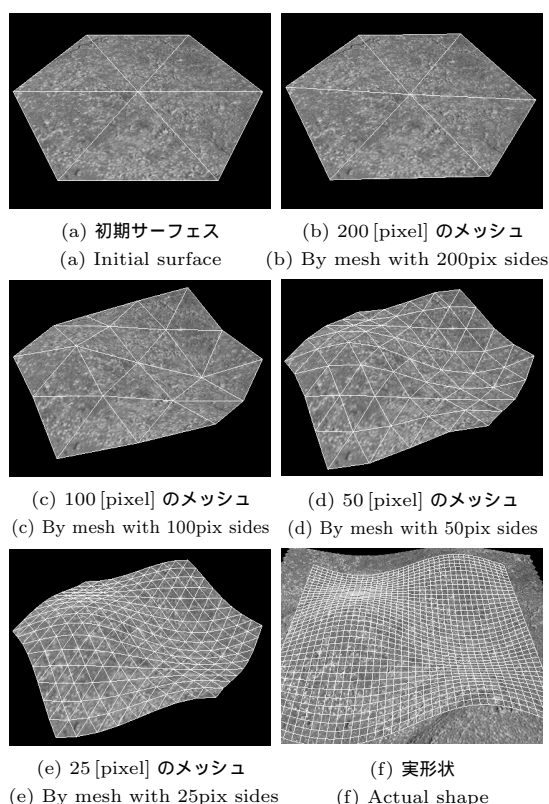


図 12 階層的メッシュによって生成されたサーフェス: $\Delta\Gamma$ の値が 10^{-4} 以下になった場合に繰返し計算を終了する収束条件を利用. 各階層における繰返し計算回数は (b) 5 回, (c) 4 回, (d) 9 回, (e) 6 回

Fig.12 Surface estimated by each meshing level: We stopped the iteration process when $|\Delta\Gamma| < 10^{-4}$. In each meshing level the iteration numbers were (b) 5 times, (c) 4 times, (d) 9 times, and (e) 6 times.

状を示し (図 11 (a) と同じ初期状態), (b) ~ (e) はそれぞれ 200, 100, 50, 25 [pixel] のパッチによって作成されたメッシュによって最終的に得られたサーフェスである. それぞれの繰返し回数は, (b) は 5 回, (c) は 4 回, (d) は 9 回, (e) は 6 回であった. 同図により, 階層的メッシュを利用することにより, 安定なサーフェス生成を実現できるだけでなく, トータルの繰返し回数を大幅に低減させた推定が可能となることが分かる.

4.1.4 計算時間の比較

本論文では, 2 章において微小平面モデルを利用したサーフェス生成を定式化し, その結果を 3 章において高速化した. そこで, その 2 つの手法の計算時間を比較し, その高速化の効果を確認する. 一方, 高速化された手法は, 対象を複数の平面によってモデル化しているものの, 対象を単一の平面と見なして 3 次

元平面パラメータを高速に推定する手法¹¹⁾ と類似した推定プロセスを用いている. その主な違いは, 提案手法では, 3 章で示したように, 繰返し計算ごとに, 式 (20) の解を求めるための H の逆行列の計算を必要とする点といえる. そこで, 参考として, 単一平面のパラメータを求める高速平面パラメータ推定¹¹⁾ の計算時間を同時に示す. ただし, サーフェス生成に利用するメッシュ全体に含まれる画素数と, 高速平面パラメータ推定の注目領域に含まれる画像は同じとする.

表 1 は, 高速化前と高速化後のサーフェス生成と, 単一平面に対する高速平面パラメータ推定手法の計算時間を比較したものである. サーフェス生成では, 4.1.1 項における実験で示した 1 辺 50 [pixel] のパッチ (頂点数 61, パッチ数 96) によるメッシュを利用した場合と, 4.1.3 項で利用した最小パッチサイズ (頂点数 217, パッチ数 384) のメッシュを利用した場合について示している. 平面パラメータで利用する ROI およびサーフェス生成の際のメッシュ内の画素数は, すべて 103,923 (約 300×300 サイズの画像と同等) である.

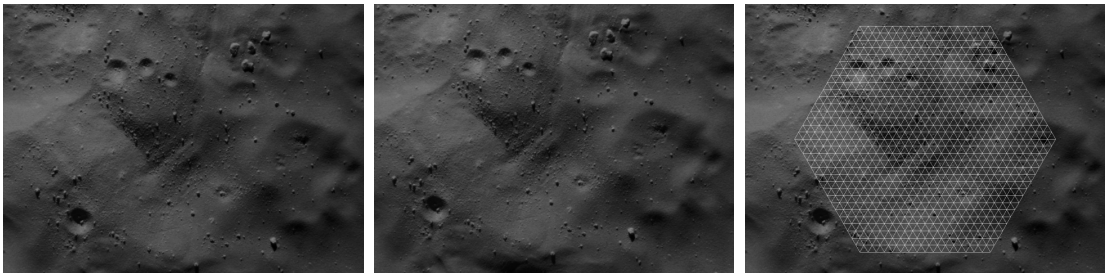
高速化前の手法 (2 章) と高速化後の手法 (3 章) とを比較すると, 高速化した手法では前処理が必要であるものの, 繰返しごとの計算時間が大幅に低減されていることが分かる. これは, 高速化前の手法では, 繰返し計算ごとに必要なヘッセ行列の計算処理の時間が大きいのに対し, 高速化した手法では, ヘッセ行列がパッチごとの疎行列の加算のみで得られ, その計算時間がほとんど無視できるほど小さいことが大きな要因である.

次に, 頂点数 61 (パッチ数 96) のメッシュによるサーフェス生成と, 単一平面に対するパラメータ推定の計算を比較すると, 理論的には逆行列のための計算時間の違いが主な違いであるのに対し, 前処理と繰返し計算とも約 2 倍の開きがある. これは, 本プログラムの実装では, メッシュ内の画素がどのパッチに含まれているかを判断する処理が含まれているためであり, その分の計算時間が加算されているためと考えられる. この様子は, 頂点数 61 (パッチ数 96) のメッシュによるサーフェス生成と, 頂点数 217 (パッチ数 384) のメッシュによるサーフェス生成では, 同じようにパッチ内画素の判断が行われているため, 前処理時間の違いがわずかであることから示される. また, 頂点数 61 の場合, 61×61 サイズの逆行列計算には, 約 1 ms の時間しかかからないが, 頂点数 217 の場合, 217×217 サイズの逆行列計算には, 約 15 ms の時間が必要となる. この違いが, 両者の繰返し計算ごとの

表 1 計算時間の比較 (単位ミリ秒): 2 章で定式化したサーフェス生成手法, 3 章で高速化した手法, および単一平面に対する高速平面パラメータ推定¹¹⁾ の計算時間を表示. サーフェス生成に利用するメッシュ全体に含まれる画素数と, 平面パラメータ推定の注目領域内の画素数を, 同じ 103,923 個としている. プログラムは C 言語で実装し, Pentium IV (2.8 GHz) の Linux OS で実行. 逆行列計算には CLapack ライブラリを利用

Table 1 Comparative study of computational time (milliseconds): We compared the computational time of the proposed surface estimation method described in Sec. 2 (non-fast), the proposed efficient method in Sec. 3 (fast), and the fast plane parameter estimation for a single plane¹¹⁾, where the numbers of pixels in the entire mesh for the surface estimation and in the ROI for the single plane estimation were the same to 103,923. All algorithms were implemented using C-language and run on a Linux PC (Pentium-IV 2.8 GHz). We used CLapack library for computing inverse matrices.

Method	Pre-computation	Per iteration	Total (5 times)	Total (30 times)
Proposed non-fast (Sec. 2) ($M = 61, N = 96$)	—	285.91	1429.6	8577.3
Proposed fast (Sec. 3) ($M = 61, N = 96$)	44.300	27.538	181.99	870.44
Proposed non-fast (Sec. 2) ($M = 217, N = 384$)	—	308.12	1540.6	9243.6
Proposed fast (Sec. 3) ($M = 217, N = 384$)	46.312	43.002	261.32	1336.4
Plane parameter estimation ¹¹⁾ (for single plane)	25.145	13.622	93.255	433.81



(a) 基準画像 (b) 参照画像 (c) メッシュ
(a) Reference image (b) The other image (c) Mesh

図 13 実画像: 月面を模擬した砂場を撮影した 1280×720 [pixel] サイズの実画像. (a), (b) はそれぞれ基準画像と参照画像. (c) のメッシュは, 階層的メッシュに用いた最も細かいメッシュを示し, 1 辺 29 ピクセルの正三角形を用いて生成したもの ($M = 817$, $N = 1536$)

Fig. 13 Real images with 1280×720 pixel size: A moon-like surface was simulated on a sandy place. (a)(b) respectively represent the reference image and the other. (c) represents the final (finest) mesh used for our hierarchical meshing approach. The mesh comprises regular triangles with 29 pixels long sides ($M = 817$, $N = 1536$)

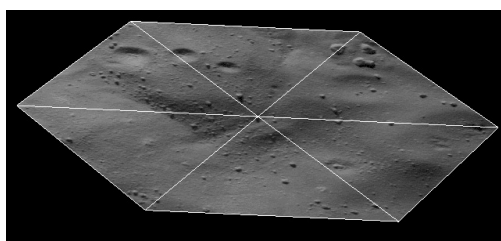
時間の差に現れており, 頂点数 (パッチ数) を増やしても, 逆行列計算以外の計算時間の差は非常にわずかなことが分かる. このことから, 提案手法は複数の平面を用いて対象のサーフェスをモデル化するものであるが, パッチ内画素の判断を無視すれば, 単一平面パラメータ推定との計算時間の差は, 逆行列計算による時間が主なものといえる.

従来のさまざまなサーフェスモデル生成法^{4), 5), 14)} が, 最低でも数分の時間を必要とすることと比較すると, 提案手法の計算時間は 1 秒程度であり, ロボットナビゲーションなどのアプリケーションにも応用可

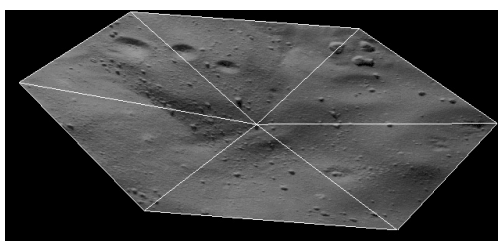
能な高速なサーフェス生成を実現できていることが分かる.

4.2 実画像実験

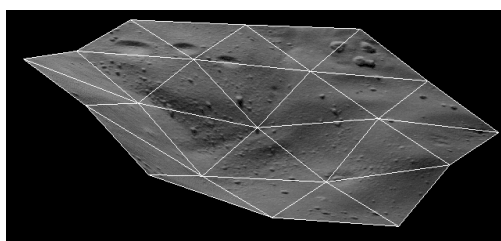
実画像実験に利用した画像を図 13 に示す. この画像は, 月表面を模擬した 1280×960 のステレオ画像である. 本実験では, 階層的メッシュによるアプローチを利用して, サーフェス生成を行った. 最初は, 画像全体を単一平面と見なし, 文献 11) による手法を用いて平面パラメータ推定を推定した. 次に, 1 辺の長さが 464 [pix] の三角パッチを用いたメッシュ ($M = 6$, $N = 7$) を生成し, 推定された平面パラメータを用い



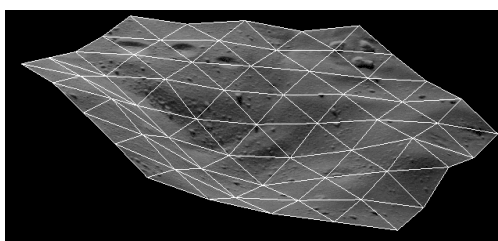
(a) 平面パラメータ推定¹¹⁾による初期平面
(a) Initial plane estimated by single plane estimation¹¹⁾



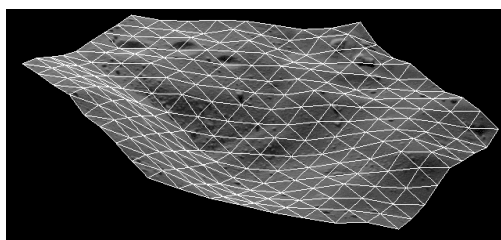
(b) 464 [pixel] のメッシュ (繰返し 10 回)
(b) By mesh with 464pix sides (after 10 iterations)



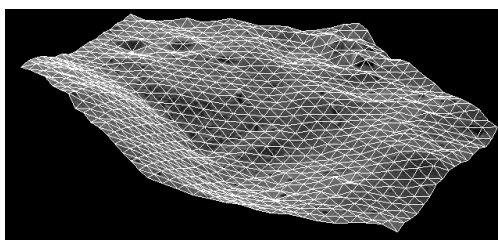
(c) 232 [pixel] のメッシュ (繰返し 17 回)
(c) By mesh with 232pix sides (after 17 iterations)



(d) 116 [pixel] のメッシュ (繰返し 4 回)
(d) By mesh with 116pix sides (after 4 iterations)



(e) 58 [pixel] のメッシュ (繰返し 3 回)
(e) By mesh with 58pix sides (after 3 iterations)



(f) 29 [pixel] のメッシュ (繰返し 4 回)
(f) By mesh with 29pix sides (after 4 iterations)

図 14 実画像に対する階層的メッシュを用いたサーフェス生成

Fig. 14 Surface estimated in each hierarchical meshing level for real images.

て、7 個の頂点の奥行き初期値を定めた。その後は、階層的メッシュを用いたアプローチにより、パッチの辺を半分にしつつ、最終的に 1 辺 29 [pix] の三角パッチによるメッシュを用いてサーフェスを生成した。最終メッシュの頂点数は 817、パッチ数は 1536 であった。

生成された表面モデルを図 14 に示す。階層的メッシュを用いたアプローチにより、最も詳細なメッシュにおいても良好にサーフェス生成ができ、メッシュ中央部に見られるエッジを持った勾配や、左上部に見られる小さなクレタの穴がおおむね良好に復元されている。

5. ま と め

本論文では、ステレオ画像を利用して、対象物体の表面形状を効率的に推定する手法を提案した。提案手法では、複数の小三角パッチによるポリゴンメッシュを用いて物体表面をモデル化し、各パッチに関する SSD

を加算した値を最小にするメッシュ全頂点の奥行きを、Gauss-Newton 法を用いて最適化した。さらに、高速平面パラメータ手法の考え方を利用することにより、効率的なサーフェス生成が実現できることを示した。

提案手法では、微小平面サーフェスモデルを対象形状にあてはめるため、対象がモデルにあてはまらない形状（不連続な形状を持つ場合や、オクルージョンなどが発生しているなど）の場合は、たとえ階層的メッシュを利用したとしても、局所解に陥りサーフェスを正しく生成できない。任意の実シーンをモデル化するためには、不連続部分やオクルージョン部分を検出し、不適切なパッチを削除するなどの手法が必要であり、その具体的な手法の検討が今後の課題である。また、ステレオ画像間でのパッチの明るさの違いや、鏡面反射の影響などを排除する方法についても検討する予定である。

謝辞 実画像実験のためのステレオ画像を提供して

くださった宇宙航空研究開発機構の片山保宏氏に深く感謝いたします。

参 考 文 献

- 1) Baker, S. and Matthews, I.: Lucas-kanade 20 years on: A unifying framework, *International Journal of Computer Vision*, Vol.56, No.3, pp.221–255 (2004).
- 2) Faugeras, O.D. and Keriven, R.: Complete dense stereovision using level set methods, *European Conference on Computer Vision*, Vol.I, pp.379–393 (1998).
- 3) Faugeras, O. and Lustman, F.: Motion and structure from motion in a piecewise planar environment, *Report de Recherche de l'INRIA* (1988).
- 4) Fua, P.: From multiple stereo views to multiple 3D surfaces, *International Journal of Computer Vision*, Vol.24, No.1, pp.19–35 (1997).
- 5) Fua, P. and Leclerc, Y.G.: Object-centered surface reconstruction: Combining multi-image stereo and shading, *International Journal of Computer Vision*, Vol.16, No.1, pp.35–56 (1995).
- 6) Gregorski, B.F., Hamann, B. and Joy, K.I.: Reconstruction of B-spline surfaces from scattered data points, *Computer Graphics International*, pp.163–170 (2000).
- 7) Hoppe, H., DeRose, T., Duchamp, T., McDonald, J. and Stuetzle, W.: Mesh optimization, *Computer Graphics (SIGGRAPH)*, pp.19–26 (1993).
- 8) Isidoro, J. and Sclaroff, S.: Stochastic refinement of the visual hull to satisfy photometric and silhouette consistency constraints, *IEEE International Conference on Computer Vision*, pp.1335–1342 (2003).
- 9) Kolmogorov, V. and Zabih, R.: Visual Correspondence with Occlusions Using Graph Cuts, Ph.D. thesis, Stanford University (2002).
- 10) Pentland, A. and Sclaroff, S.: Closed-form solutions for physically based shape modeling and recognition, *IEEE Trans. Pattern Analysis and Machine Intelligence*, Vol.13, pp.715–729 (1991).
- 11) 杉本茂樹, 奥富正敏: ステレオ画像を用いた高速な平面パラメータ推定法, 情報処理学会論文誌: コンピュータビジョンとイメージメディア, Vol.48, No.SIG 1(CVIM 17), pp. 24–34 (2007).
- 12) Szeliski, R. and Tonnesen, D.: Surface modeling with oriented particle systems, *Computer Graphics (SIGGRAPH)*, Vol.26, pp.185–194 (1992).
- 13) Yokoya, N.: Surface reconstruction directly from binocular stereo images by multiscale-multistage regularization, *IEEE International Conference on Pattern Recognition*, pp.642–646 (1992).
- 14) Zhang, L. and Seitz, S.: Image-based multiresolution shape recovery by surface deformation, *SPIE: Videometrics and Optical Methods for 3D Shape Measurement*, pp.51–61 (2001).

(平成 19 年 1 月 10 日受付)

(平成 19 年 7 月 31 日採録)

(担当編集委員 植芝 俊夫)



杉本 茂樹

1995 年東京工業大学制御工学卒業。1997 年東京工業大学大学院情報理工学研究科情報環境学専攻修士課程修了。2003 年東京工業大学大学院理工学研究科機械制御システム専攻産学官連携研究員。コンピュータビジョン, 画像計測, ITS 関連の研究に従事。電子情報通信学会, IEEE 各会員。



奥富 正敏 (正会員)

1981 年東京大学工学部計数工学科卒業。1983 年東京工業大学大学院理工学研究科制御工学専攻修士課程修了。同年キャノン (株) 入社。1987 年～1990 年カーネギーメロン大学コンピュータサイエンス学科客員研究員。1994 年東京工業大学大学院情報理工学研究科情報環境学専攻助教授。2002 年同大学院理工学研究科機械制御システム専攻教授。コンピュータビジョン, 画像処理, 画像計測に関する研究に従事。工学博士。電子情報通信学会, 計測自動制御学会, 画像電子学会, IEEE 各会員。