社団法人 電子情報通信学会
THE INSTITUTE OF ELECTRONICS,
INFORMATION AND COMMUNICATION ENGINEERS

信学技報
TECHNICAL REPORT OF IEICE

2002 年 PRMU アルゴリズムコンテスト 「砂嵐から立体を見つけ出そう」 実施報告とその入賞アルゴリズムの紹介

山澤 一誠[1]　石田 皓之[2]　岡本 崇弘[3]　小田 昌宏[2]
前橋 久美子[3]　浅井 俊弘[1]　牧田 孝嗣[1]

[1]奈良先端科学技術大学院大学 〒630-0192 奈良県生駒市高山町 8916-5
[2]名古屋大学　[3]広島市立大学大学院
E-mail: [1]yamazawa@is.aist-nara.ac.jp

あらまし　パターン認識・メディア理解（PRMU）研究専門委員会は，平成 9 年より研究活動の一環として，若手研究者の発掘・育成を目的としたアルゴリズムコンテストを実施している．第 1 回目のアルゴリズムコンテストではランダムドットステレオから視差画像を求めるコンテストを行った．第 6 回目の今年はその第 1 回目のテーマに立ち戻り，さらに多眼というアレンジを加え，多眼ランダムドットステレオをテーマとしてアルゴリズムを募集した．ステレオ画像の枚数に応じて３つのレベルを設定し，ひとつのアルゴリズムによる複数のレベルへの応募も可能とした．募集の結果，18 アルゴリズム，のべ 34 件の募集があり，審査により 6 アルゴリズムを入賞とした．さらに情報科学技術フォーラム（FIT2002）において審査結果発表と表彰式，入賞者によるアルゴリズムの紹介，第 1 回アルゴリズムコンテスト最優秀賞受賞者による特別講演を行った．本報告ではそれらの実施報告と入賞者によるアルゴリズムの紹介をする．

キーワード　アルゴリズムコンテスト，多眼ランダムドットステレオ

2002 PRMU Algorithm Contest ”Let’s find objects from random-dots-images” Summary Report and Its Prize Winning Algorithms

Kazumasa YAMAZAWA[1]　Hiroyuki ISHIDA[2]　Takahiro OKAMOTO[3]　Masahiro ODA[2]
Kumiko MAEBASHI[3]　Toshihiro ASAI[1]　Kohji MAKITA[1]

[1]Nara Institute of Science and Technology 8916-5 Takayama, Ikoma, Nara, 630-0192 Japan
[2]Nagoya University　[3] Hiroshima City University Graduate School
E-mail: [1]yamazawa@is.aist-nara.ac.jp

Abstract　The Special interest Group on Pattern Recognition and Media Understanding (PRMU) has been promoting an annual algorithm contest on pattern recognition and media understanding since 1997. The first contest focused on estimating depths of random-dots-stereogram. The 6th this year contest returns to the first theme, adds the arrangement, and focuses on estimating depths of random-dots-images of multi-stereo. The problems of the contest are divided into the 3 levels according to the number of stereo images. And, it is possible to enter two or three levels by one algorithm. As a result of the invitation, 34 entries of 18 algorithms were accepted. Then, the jury considered 6 algorithms as winning prize. In Forum on Information Technology (FIT2002), PRMU performed the announcement of the examination result, the commendation ceremony, the presentations of the algorithms of the prizewinners, and the special lecture of the highest award winner of the 1st algorithm contest. This report describes the summary of the contest and the details of the prizewinners’ algorithms.

Keyword　algorithm contest,　multiple random-dots stereo

1. はじめに

パターン認識・メディア理解（PRMU）研究専門委員会は,平成9年より研究活動の一環として,若手研究者の発掘・育成を目的としたアルゴリズムコンテストを実施している．第1回目のアルゴリズムコンテストではランダムドットステレオから視差画像を求めるコンテストを行い，毎年異なるパターン認識・メディア理解のテーマでコンテストを行ってきた．第6回目の今年は第1回目のテーマに立ち戻り，さらに多眼というアレンジを加え，多眼ランダムドットステレオをテーマとしてアルゴリズムを募集した．結果，18アルゴリズムの募集があり，審査により独創的な6アルゴリズムを入賞とした．さらに9月25日の情報科学技術フォーラム（FIT2002）において審査結果発表と表彰を行った．表彰式では横矢実行委員長の挨拶のあと，審査結果の発表，表彰式，特別講演，入賞者によるアルゴリズムの紹介を執り行った．表彰式では原知的総合通信システム基金の湊常務理事より入賞者に表彰状，トロフィー，副賞が贈呈された．また，特別講演では今回のテーマと似たテーマで行った第1回アルゴリズムコンテストの最優秀賞受賞者である航空宇宙技術研究所の片山様より，「アルゴリズムコンテストに参加して」と題して講演を行っていただいた．片山様のアルゴリズムコンテストに参加するきっかけ，最優秀賞のアイデアを出すきっかけ，それから現在までの経緯などを聞くことができ，今回の入賞者はもとより他の方にも非常に興味のある講演であった．

2. 課題

本コンテストでは，パターン認識・メディア理解に関する基本的な問題を出題し，同分野に対する興味と理解を高めるとともに，これからの時代に貢献できる創造的な若手研究者の発掘・育成を目的としている．そのため，高専，大学，大学院の学生を想定した出題となっている．本年度のテーマは第1回目のテーマに立ち戻りさらにアレンジを加えた．人間は右目と左目しか持たないが，機械は2眼だけでなく必要ならば多くの目を持つことができる．今回はこの機械特有のアドバンテージである多眼ステレオについてのアルゴリズムを募集した．ただ，問題の簡単化のため画像は第1回目と同様にランダムドットステレオにした．そこで「砂嵐から立体を見つけ出そう」とキャッチコピーをつけてアルゴリズムを募集した．

今回の課題はステレオ入力枚数に応じた３つのレベルを用意し，それぞれのレベルごとに応募アルゴリズムを審査した．３つのレベルすべてに対応したアルゴリズムのように複数のレベルに応募するのも可能とした．提出するプログラムはＣ言語に基づく関数で，２値ランダムドット画像（0or255の２階調）を入力とし，それを処理して視差（距離）を計算する．各レベルは以下の通りである．

- レベル１：２眼ステレオ
  左と右の２枚の画像を入力とする．
- レベル２：３眼ステレオ
  中心と右，上の３枚の画像を入力とする．
- レベル３：９眼ステレオ
  3x3に配置した９枚の画像を入力とする．

3. 応募と結果

18件の応募があった．レベル別の応募件数ではレベル1に14件，レベル2に8件，レベル3に12件となった．各レベルの応募件数の合計ともとの応募件数が異なるのは1件の応募で複数のレベルへのエントリーが可能なためである．審査のデータとして単純なものから複雑なものまで9種類の奥行き画像に対応する多眼ランダムドットステレオ画像を各アルゴリズムに入力し，処理時間と精度を計測した．これらの処理速度と精度，さらに各応募者が書いたアルゴリズムの説明を19人の審査員に見てもらい,各アルゴリズムにつき10点満点で主観評価で点数をつけてもらった．この点数を集計し，上位５位のアルゴリズムを入賞とした．この入賞者の中より各審査員の投票により最優秀と優秀賞を一人ずつ決定した．また，最もアルゴリズムが複雑になるレベル３で精度が最も高かったアルゴリズムを特別賞とした．

最優秀賞：石田 皓之 殿
（名古屋大学電気電子情報工学科
情報工学コース3年）
優秀賞：岡本 崇弘 殿
（広島市立大学大学院情報科学研究科1年）
入賞：小田 昌宏 殿
（名古屋大学電気電子情報工学科
情報工学専攻3年）
前橋 久美子 殿
（広島市立大学修士1年）
浅井 俊弘 殿
（奈良先端科学技術大学院大学
情報科学研究科修士1年）
特別賞：牧田孝嗣
（奈良先端科学技術大学院大学
情報科学研究科修士1年）

4. 応募アルゴリズム

応募されたアルゴリズムには基本的なブロックマッチングを用いたアルゴリズムが多かったが，ブロックマッチングに独創的な工夫を加えたものや，またはブロックマッチングを用いない独創的な手法があった．以降の章は受賞者による各アルゴリズムの説明だが，これらのアルゴリズムは審査員に主観評価によって判断された，独創的なアルゴリズムである．これらの説明は各受賞者から異なる形式で提出していただいたものであるため，多少体裁に差異があることをご容赦いただきたい．

謝辞

（協賛） 原総合知的通信システム基金，NTT コミュニケーション科学基礎研究所，富士通，三菱電機，東芝，松下電器産業，日立製作所，日本電気，豊田中央研究所，NTT DoCoMo，NTT データ（順不同）

実行委員長： 横矢直和
（奈良先端科学技術大学院大学）

実行委員： 山澤一誠
（奈良先端科学技術大学院大学）
椋木雅之（京都大学）
柴田史久（大阪大学）
安村禎明（東京工業大学）
神原誠之
（奈良先端科学技術大学院大学）
村瀬洋（NTT コミュニケーション科学基礎研究所）
直井聡（富士通）
馬場口登（大阪大学）
佐藤真一（国立情報学研究所）
角保志（産業技術総合研究所）
磯俊樹（NTT DoCoMo）
入江文平（東芝）
池田尚司（日立製作所）
坂野鋭（NTT データ）
長尾健司（松下電器産業）
二宮芳樹（豊田中央研究所）
橋本学（三菱電機）
山田敬嗣（日本電気）
(順不同)

協力者： 佐藤智和
（奈良先端科学技術大学院大学）

5. 最優秀賞アルゴリズム紹介

名古屋大学工学部 電気電子情報工学科
情報工学コース 3 年 石田 皓之

1 アルゴリズム概要

今回のアルゴリズムは、ランダムドット解析のときに同じ視差が連続して求まるという性質があることを利用して、視差計算の回数を可能な限り減らすことを第一の目標として作成した。

通常、視差の計算には視差を求める点を中心とした 3×3 またはそれ以上の大きさの領域にある点を調べ、マッチングを行うが、ここでは最大でも 3×3 の 9 点のみの計算に抑えられるようにし、可能な限り、その 1 点のみから求められるようにした。

計算時間の短縮を最優先としながらも、誤差が大きくならないように、誤りをしたことがわかったときに後退して修正する機能、視差が求められるランダムドット入力画像をある程度計算によって選択する機能を追加した。

2 アルゴリズム内容

このアルゴリズムは画像の中心から外側に向かって渦巻き状の軌道で反時計回りに解析をする。解析中の各点において、1 点前の点と、進行方向に向かって左手側 (内側) の点は解析済みであり、解析中の点の視差を求めるための参考にすることができる。求める視差は近辺の視差に等しい場合が多いので、それらの視差は有用な情報となる。

解析の方法には 3 種類あり、その解析する点だけを、一定の入力画像のみで判断する方法 (関数:check_dot) 、解析する点を中心とし、3×3 の範囲で調べ、パターンがどれだけ合うかを全ての視差について調べる方法 (関数:check_box)、視差を予想し固定して、全ての入力画像に対して 3×3 の範囲でそれぞれの方向に調べる方法 (関数:check_shade) に分かれる。

視差は線形リストを用いて、最近得られた解析の結果から、新しい順に調べるようにした。視差が求まる度にその視差をリストから除き、リストのトップに加える。同じ視差が連続して現れる場合が多いので、このような方法を採用している。

各点の解析は 1 点解析から行う。視差は 1 点前の視差と等しいものと予想してランダムドット入力画像から点が合うかどうかを調べる。合えばその時点で視差は決定となる。合わない場合、同様に進行方向に向かって左手側の点を参考にして 1 点解析を行う、これも合えばその時点で視差は決定となる。これらの解析は、進行方向に対して左手側に平行移動を行ったランダムドット入力画像を比較の対象とする。

合わなかった場合は、視差を視差リストのトップにある視差だと予想して、関数 check_shade によって解析を行う。その部分のパターンが平行移動による上書きによって消えたものでないかを確かめるためである。3×3 の範囲内で全てのランダムドットのパターンが等しい入力画像が 1 つでも見つかったとき、解析結果はその視差で決定となる。

それでも合わなかった場合、次は関数 check_box によって最も確からしい視差を求める。3 × 3 の範囲で調べ、視差はリストの順に整合するかどうか調べる。このとき、比較するランダムドット入力画像は、その調べる高さが計算済みの内側の視差より小さかった場合は、(パターン上書きの影響がないようにするため) 進行方向に対して左手側に平行移動を行ったランダムドット入力画像を比較の対象とする。それ以外の場合は、進行方向に対して右手側に平行移動を行ったランダムドット入力画像を比較の対象とする。途中で 3 × 3 の範囲内全てのパターンが一致する視差が見つかれば、その時点で視差はその値で決定となる。

このアルゴリズムでは、視差の境界地点で誤りを起こしやすいので、後退して誤りを訂正する処理を加えた。1 点前の点の解析結果と内側の点の解析結果が異なり、1 点前の点が内側の点の視差でもマッチする場合、誤りを起こしていた可能性がある。この誤り訂正は修正個所の点がその修正された新しい視差でも正しい (1 点判定 check_dot を用いる) 間は後退して繰り返される。

このとき 1 点単位で調べると、再び逆方向に誤る可能性があるので、その内側の点と比較して、誤りがないようにする。後退修正が終わった後は、後退する前に解析していた点から解析を再開する。

毎回の 1 点判定の度に、その決定が誤りでないか確認するようにすれば、このような修正は行う必要はない。しかしそれでは非常に時間がかかる。間違いが発見される回数は全体を調べる回数に比べると少ないので、通常時は最低限のチェックで読み飛ばすようにし、間違いがわかった時点で後退して訂正するようにした。この処理を行うことで、視差の境界付近の精度は改善された。

3　処理結果と考察

処理結果の表は次のようになった。(このアルゴリズムはレベル 3 のみを対象としている)

	Level3 エラー率	Level3 処理時間
最小	0.02	0.08
平均	3.32	0.12
最大	24.82	0.29

今回作成したプログラムでランダムドット解析を行った場合、画面上に現れる視差の種類が比較的少なく、与えられた画像中の図形が単純な形状である場合、計算時間が非常に短く、精度も良くなっている。一方、解析中に調べる点の視差が頻繁に変わるような画像や、視差がなだらかに変化する部分を含む画像では、計算時間が長くなると同時に、正答率も大幅に低下している。

カメラからの取り込んだような複雑なランダムドット画像は、仮に広い範囲の領域でマッチングを行っても正確な結果は得られないといえる。もしこのような画像でも高い正答率を得ようと思ったら、確かな視差が求められる部分から求めていき、複雑な部分はその周辺で確かな視差が求められている部分から計算して求めるなど、全く別の手法をとらなければいけないのではないかと考えられる。

今回は計算時間の短縮を優先にプログラムを組み立てたが、それでも正答率を低下させないためにある程度時間をかけなければいけない処理もあり、正答率と計算スピードのどちらをとるかという選択は難しかった。また誤りの修正には後退修正という手段をとっているが、もし画像に細い領域が存在していたとすれば、修正が行われる前にその領域をまたいでしまい、修正できなくなる場合がある。すると本来 1 つの図形であるものが分断されてしまう。このような誤りは、それほど正答率の低下にはつながらないが、形状が変化してしまうことになるので、他の場合の画像認識には応用できない。そのためこの手法は今回のコンテストにしか使用できないという問題も持っている。

6. 優秀賞アルゴリズム紹介

氏名：岡本 崇弘 (オカモト タカヒロ)
所属：広島市立大学大学院情報科学研究科
知能情報システム工学専攻
情報認識学講座 1 年

1　アルゴリズムの概要

複数枚のランダムドット画像から視差を計算する方法として，各画像間で視差分だけずらした差分画像を生成した際に現れる差分のない領域の面積を評価値として与える手法を提案する．

生成された差分画像は，図 1 のようにその視差における正解領域である差分がない黒く塗りつぶされた領域と，その他のランダムドット領域に分割することができる．差分画像における各画素が正解領域であるかランダムドット領域であるかを判断するために，評価値を設ける．この評価値は注目画素を中心に広がる差分がない領域の面積とする．この面積が広い場合は注目画素が正解領域に含まれている可能性が高いと考えられる．また，面積が狭い場合や注目画素自体に差分がある場合は注目画素はランダムドット領域に含まれる可能性が高いと考えられる．この操作を各組の差分画像における視差 0〜50 の場合に対して行い，最も評価値が高いときの視差を結果として格納する．

これらの操作をフローチャートにより示したものを図 2 に示す．

1.1　正解領域検出アルゴリズム

本手法では画素の評価を行う際に，視差画像における差分がない領域の面積に着目している．この面積を計算する際に，連続するすべての差分がない画素を計算する方法が考えられるが，この場合，ランダムドット領域であっても差分のない領域が深くまで継っている可能性がある．そこで，ランダムドット領域における評価値を抑制するために，ここでは次のような方法を用いる．

まず注目画素を中心に x 軸方向 (左右方向) に連続する正解をカウントする．次にカウントされた各画素に対して y 軸方向 (上下方向) に連続する正解をカウントする．例えば，図 3 において，注目画素を中心とした x 軸方向には 6 カウント，さらにカウント

図 1: 差分画像

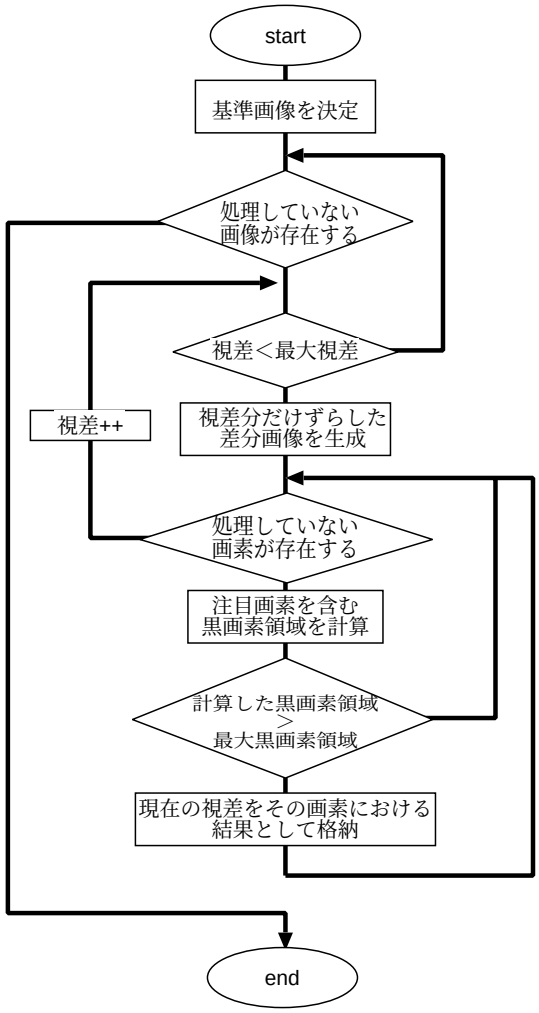

図 2: フローチャート

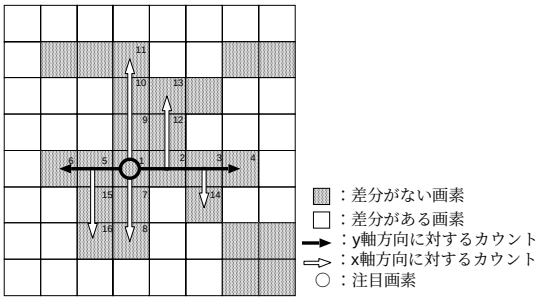

図 3: 画素の評価方法

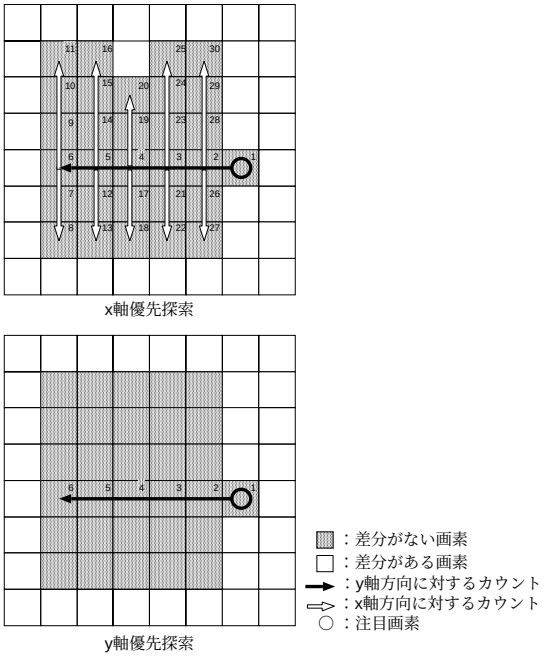

図 4: カウント数の低い y 軸優先探索を評価値として与える

された各画素の y 軸方向に 0，2，5，2，1，0 の合計 16 カウントが注目画素の評価値として与えられる．

この操作を x 軸と y 軸を入れ換えて同様に行う．x 軸を最初に検索する場合と y 軸を最初にカウントする場合のどちらを優先して評価値とするかについてであるが，エッジ付近における精度向上のために，低い方の評価値を与える．例えば，図 4 のような状況のとき，x 軸優先では 30，y 軸優先では 6 と大きく異なった評価値となる可能性がある．このような場合，注目画素はランダムドット領域が偶然正解領域に含まれてしまっている可能性が高いので，低い方の評価値である y 軸優先によるカウントを与える．これにより評価値の抑制ができ，より正しい値が適用される．

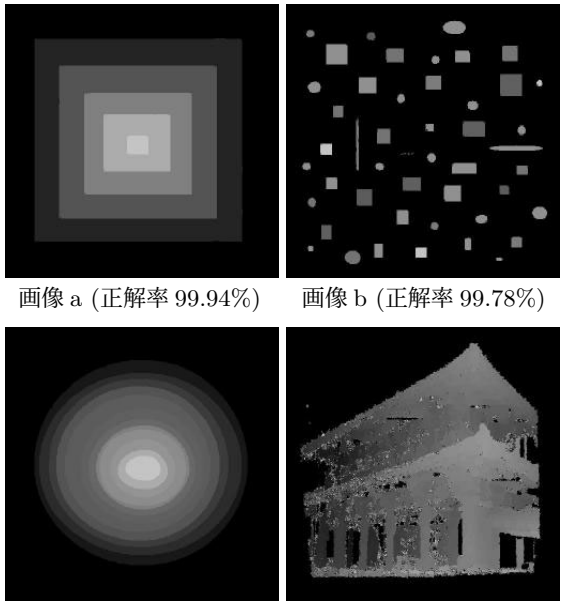

図 5: 処理結果

2　処理結果と考察

図 4 に生成した画像の例と正解画像との比較を行った際の正解率を示す．画像 (a)〜(c) のように各視差の領域がある程度大きさを持った画像に対しては 90%後半の正解率が得られ, 比較的高精度な結果となったが，画像 (d) のように各視差の領域が細かく分割されている画像に対しては十分な結果が得られたとは言えない．これは視差を計算する際に差分のない領域の面積を評価値として与えているため，正解領域が微小である画像に対しては適切な結果が得られないためであると考えられる．

3　まとめ

今回は，複数枚のランダムドット画像から視差画像を生成するために，ランダムドット画像の差分をとり，一致した領域の面積を計算するとこで高精度な視差画像の生成を行った．また，領域を探索する範囲に制限を設けることでエッジ付近における精度の向上を実現した．

この手法を用いることで，多くのランダムドット画像からの視差計算において，高い精度で視差を求めることが可能であった．

7. 入賞アルゴリズムの紹介

小田 昌宏
名古屋大学工学部電気電子・情報工学科
情報工学専攻　3年

1.　アルゴリズムの概要

アルゴリズムはレベル1，2，3すべてに対応したものとなっている。

視差を求めるため4つのウィンドウ（画像の比較範囲）を使用し、いずれかのウィンドウ内の画像が完全に一致したときに視差を採用する。

処理の高速化のため、すでに視差がもとめられた隣の画素の視差を利用して視差を推定する。

視差の計算後、視差が発見されなかった部分を対象に、4近傍の膨張、収縮処理を各1回ずつ行う。

2.　アルゴリズムの内容

2．1　視差を求める処理

入力画像はレベルごとに違う枚数の画像が与えられるが、どのレベルでも中心の画像（0番の画像）と他の画像を2枚ずつ組にして画像比較を行い、視差を求める。

2枚の画像を比較する際には、4つのウィンドウを使用する。これを図1に示す。

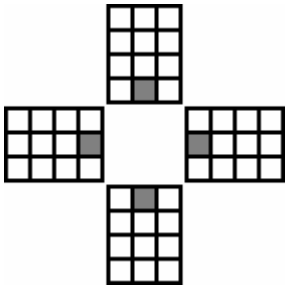

図1　4つのウィンドウ
（グレーの部分が注目画素）

ウィンドウ内の画像を比較しながら0から49までの50通りの視差について調べる。4つのウィンドウのいずれかでウィンドウ内の画像が完全に一致していれば、注目画素の視差として採用する。0から49までの視差を順に調べる中で複数の視差が発見された場合は、最後に発見された視差を採用している。（これはプログラムのミスであり、本来は最初に発見された視差を採用するつもりだった）

2．2　処理の高速化

画像の全ての画素に対して上記の視差を求める処理を行っていると、処理時間が長くなってしまう。そこで、画像の隣り合う画素の色は同じことが多いという特徴を利用して処理の高速化を図った。

視差を求める処理は画像の上から下へ一行ずつ、行の中では左から右へ向かって順に視差を求めていくので、注目画素の左と上にあるそれぞれ1画素は既に視差が求まっている。これを利用し、注目画素と、その左と上の画素の視差が同じでないか調べる処理を、視差を求める処理の前に行った。この高速化処理によって視差が発見されれば、これを注目画素の視差として採用し、次の画素の処理に移る。

視差を求める処理は50通りの視差についてチェックするのに対し、この高速化処理は1通りの視差をチェックするだけなので、これにより視差が発見されれば高速に処理を進めることができる。

2．3　ノイズ消し処理

これまでに説明した視差を求める処理と高速化処理により求められた視差画像の例を図2に示す。白い部分の中にノイズのような多数の点が入っている。白い部分は本来視差の求まらない部分であり、ノイズのような点は誤って視差が求められた部分である。

ノイズのような点を消すため、白い部分を対象に4近傍の1画素膨張処理を行い、次に4近傍の1画素収縮処理を行った。これによりノイズのような点をかなり消すことができる。図2の画像にこの処理を行った結果は図3のようになる。

図2 ノイズ消し処理前

図3 ノイズ消し処理後

２．４　全体の処理の流れ

アルゴリズムの全体の流れは以下のようになる。括弧内は処理の説明をしている章番号である。

ある注目画素に対して

１．高速化処理を行う（2.2)。これで視差が発見されれば次の画素の処理へ移る。

２．視差を求める処理を行う（2.1)。

上記の処理を繰り返して全ての画素の視差を求めた後、ノイズ消し処理（2.3）を行う。

3.　処理結果と考察

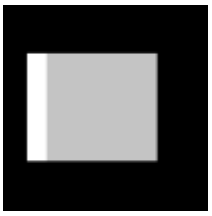

図４　正解画像の例１

図５　正解画像の例２

正解画像が図 4 のような単純な画像の場合は処理の高速化が効果的に働いて高速に視差を計算することができる。この画像の処理結果はエラー率 0.05%、計算時間 0.43 秒となった。

図 5 のような濃淡変化の激しい画像では隣同士の画素の視差が異なることが多く、処理の高速化があまり働かない。さらに、ノイズ消し処理によって画像の白い部分が膨張し、正しい視差の求められている点まで消してしまうことがある。これらのことから、アルゴリズムはこのような画像には不向きである。この画像の処理結果はエラー率 22.1%、計算時間 2.09 秒となった。

4.　まとめ

このアルゴリズムは、求まる視差画像が一つの色で広範囲を塗りつぶしたような単純な画像の場合には高速かつ正確に視差を計算することができる。視差を求める処理は、画像の中で視差が本来求まらない部分で誤った視差を求めてしまうことが多いが、これをノイズ消し処理で補っている。

求まる視差画像が画素ごとの濃淡変化の激しい、写真のような画像の場合は視差の計算に時間がかかってしまう。さらにノイズ消し処理によってエラー率が増えてしまうこともある。

さらによいアルゴリズムにするためには、入力画像の特徴を調べ、それによって視差を求める処理のウィンドウの範囲を変化させることや、場合によってはノイズ消し処理を行わないようにするなどの改善点が考えられる。

視差を計算する処理の中で、0 から 49 までの全ての視差を調べて最後に発見された視差を採用しているのはプログラムのミスであり、本来は最初に発見された視差を採用して、それ以降のものは調べないようにするつもりだった。(for ループの中の break が continue になっていた) このミスを修正すれば処理を少し高速にすることができる。

8. 優秀賞アルゴリズム紹介

前橋 久美子
(広島市立大学大学院 情報科学研究科)

1　アルゴリズムの概要

精度向上および，高速化の二点を目標に通常のテンプレートマッチングの問題点を考察した．まず精度について，奥行きが変化する部分と画像の端の部分で正しい視差を求めるのが難しく，精度を低下させていることがわかった．次に，速度についてであるが，背景部分でのテンプレートマッチングの時間が無駄であると考えられる．以下この 2 点を改善するアルゴリズムについて述べる．

2　アルゴリズムの内容

今回考えたアルゴリズムの流れを図 1 に載せる．

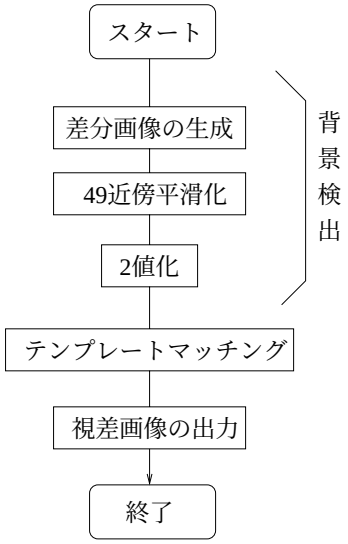

図 1: アルゴリズムの流れ

図 1 に示したように，無駄な背景部分の処理を省くために，まず背景検出を行う．そして，奥行きが変化する部分と画像の端の部分での精度低下問題に対応したテンプレートマッチングを行い，視差画像を得る．以降，2.1 節で背景検出について述べ，2.2 節でテンプレートマッチングについて説明する．

2.1　背景検出

まず背景検出部分では，**1) 差分画像生成 2) 24 近傍平滑化 3) 二値化** の 3 つの処理を行う．ここでは 2 枚の画像を用いて背景検出を行う．

まず始めに，2 枚の画像の対応する画素で輝度を減算することにより差分画像を生成する．

差分画像では，背景以外の部分でも偶然輝度が一致してしまう個所が発生する．この部分を背景としてしまわないように，49 近傍平滑化を行う．この処理は, 注目画素を中心とした 7×7 画素の輝度の平均値を注目画素の輝度とするものである．

次に平滑化した画像において，黒以外の画素を白とすることで 2 値化する．

以上の処理により，背景部分が黒，それ以外が白であるような画像が得られる．

2.2　テンプレートマッチング

背景検出部で得られた画像の背景以外の部分に対してテンプレートマッチングを行う．

一般にテンプレートマッチングは基準となる画像から，対応点を求めたい画素 (注目画素) を中心としたブロック状の部分画像を切り出し，他のカメラから得られた画像で最も誤差の低いブロックを求め，その中心点を対応点とするというものである．こうして求めた対応点と注目画素との距離を視差とする．ここで，誤差とは基準となる画像のブロックと，対応点を探索する画像のブロックで，対応画素の輝度が異なるものの数である．

しかし，ブロックの中心に注目画素を持ってくると，奥行きが変化する部分と，画像の端の部分では正確な視差が求まりにくい．そこで，より正確に視差を求めるために，注目画素を中心からずらしたようなブロックを提案する．テンプレートマッチングに用いたブロックを図 2 に示す．色のついた画素が注目画素を表す．

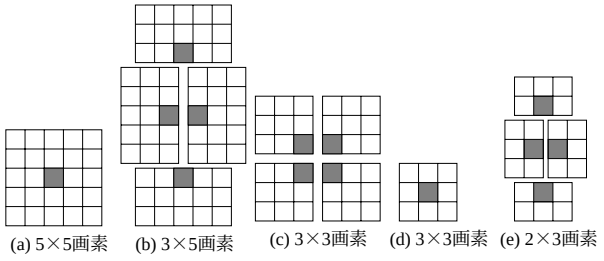

図 2: ブロックの種類

テンプレートマッチングでは，ブロックが小さすぎ

ると偶然一致してしまう個所が発生するため，正確な視差が求められなくなる．そこで，図 2(a) の大きいブロックでほぼ正確に視差が求められる画素に対してはこのブロックを用いる．(a) で求められなかった画素に対しては，(b)-(e) と順々に小さなブロックを用いて視差を求めていく．

ここで，(a),(e) のような 1 枚のブロックを用いてマッチングを行う場合と，(b),(c),(e) のように 4 枚のブロックを用いてマッチングを行う場合の対応点の決定方法について述べる．以降，9 眼ステレオの中心のカメラから得られる画像を基準画像とする．

1 ブロック 基準画像とそれ以外の画像との誤差の和を求める．誤差の和が最小となる位置を求め，そこでの総誤差が閾値以下であればそこを対応点とする．

4 ブロック 1 つのブロックで，基準画像とそれ以外の画像の誤差を求め，求めた誤差が閾値以下である画像の数をカウントする．この処理を 4 つのブロックそれぞれについて行い，4 つの誤差が閾値以下の画像数を得る．それらの値のいずれかが 2 以上であれば，そこを対応点とする．

また，どのブロックを用いても視差が求まらない画素に関しては視差を -1 とする．

また，実画像のような奥行きの変化の激しい画像では視差がうまく求まらないため，ブロック内の画素に重みづけをした．各ブロックの重みは中心画素を $(0, 0)$ として $5 - (|x| + |y|)$ で求めた．

これらの手法により，視差の境界に強く，また様々な画像の性質にも柔軟に対応できると考えられる．

3 結果

各レベルにおける提案手法の結果を以下の表 1 に示す．

表 1: 提案アルゴリズムの精度と実行時間

		レベル 1	レベル 2	レベル 3
エラー率 (%)	最小	2.71	0.40	0
	平均	6.66	3.81	2.30
	最大	24.84	22.85	18.24
実行時間 (sec)	最短	5.02	6.81	23.97
	平均	11.64	22.20	87.86
	最長	19.78	37.01	143.35

4 まとめ

視差が一定である領域がある程度大きいような画像では良い結果が得られた．特に，視差の境界が直線的であるような人工画像では 100%に近い精度が得られた．しかし，実画像のように視差が一定となる領域がごく狭い画像では，視差が一定である領域よりもブロックの方が大きくなるため，精度が上がらなかった．この点はブロックの重みを変更すれば，もう少し良い結果が得られると期待できる．

実行時間は多くのブロックを使って，処理が複雑になったために長くなってしまった．しかし，背景切り離しの処理自体はほとんど時間がかからない．従って今回の問題のように背景の面積が大きい場合には，この処理は有効である．

9. 入賞アルゴリズムの紹介

浅井　俊弘
奈良先端科学技術大学院大学　博士課程前期課程１年

1. アルゴリズムの概要

人間は目を 2 つしか持っていないが、機械は必要に応じて複数のカメラを備えることができる。そのアドバンテージを活かして立体認識を行う。

本アルゴリズムは、カメラ位置の異なる 9 枚の画像を用い、その画像から視差を推定する際に行うテンプレートマッチングを効率的に行うことで処理の高速化を図るものである。また、マッチング時のウインドウ間誤差の計算に重みを持たせることで精度の向上も図る。

・高速化の方法

a) 予めウインドウ内のデータを 1 つの変数として格納しておきマッチングの時間を短縮する

b) マッチング時にウインドウ間の誤差が 0 となった時点でウインドウ内の全画素の距離を確定する

・エラー率を下げる方法

c) ウインドウの中心に大きな重み付けをする

2. アルゴリズムの内容

a) 予めウインドウ内のデータを1つの変数として格納しておき、マッチングの時間を短縮する

予め 1 つのウインドウ内のデータを 1 つの変数として格納しておくことで、ウインドウの比較の際その変数を 1 回比較するだけで誤差の計算を可能にする。本プログラムではウインドウサイズが 5×5(25bit)を用いた。その 25bit 分のデータを 1 つの int 型(32bit)の変数に格納する。

b) マッチング時にウインドウ間の誤差が 0 となった時点でウインドウ内の全画素の距離を確定する

マッチングではカメラ位置が中心の画像と他のカメラ位置での画像との誤差を求めるが、1 つでもウインドウ間の誤差が 0 になればそれ以上良い結果が出ることは無いので、誤差が無かった時点で他のカメラ位置での画像との比較を中断する。また、あるウインドウ内で異なった距離が含まれていれば、カメラ位置が変わったときにずれ方が違うので同じランダムドットパターンにはならずウインドウ間に誤差が生じるはずである。つまり誤差が無いということはそのウインドウ内全てが同じ距離と考えることができるので注目画素の周り 24 近傍も注目画素とおなじ距離にする。このことで一度に複数の画素の距離を確定できるので処理時間が短縮する。

c) ウインドウの中心に大きな重み付けをする

入力画像にノイズが含まれないと仮定すると、対応している画素同士ならばウインドウの中心(注目画素)は必ず同じになる。逆に言えば、誤差は小さくても中心が違えば対応点でない可能性が高い。そこでウインドウの中心の重みを大きくしてウインドウ間の誤差の計算を行うことで誤った視差の推定を減らす。

3. 処理結果

処理結果を以下に示す。

表 1 エラー率[%]

最小	平均	最大
0.15	2.31	14.39

表 2 計算時間[sec]

最小	平均	最大
3.73	5.29	13.43

4. まとめ

マッチングの高速化処理とウインドウの重み付けによってランダムドットから立体を認識するアルゴリズムを提案した。結果が複雑な画像でもエラー率を改善することができた。

エッジ部分ではマッチング時に対応点にも関わらず誤差が大きくなる場合がある。このことを考慮に入れることでさらにエラー率を下げることができると考えられる。

10. 特別賞アルゴリズムの紹介

牧田　孝嗣

奈良先端科学技術大学院大学

博士前期課程１年

はじめに

本手法では、level3 において 3×3 に配置された９枚のランダムドットステレオ画像を入力とし、中心の画像における視差を高速に精度良く計算する事を目標とする。ただし、プログラムの製作においては、高速に計算を行うことよりも精度の高さを重視し、level3 における精度の限界を目指す。

視差計算の概要

1. 視差の候補となる値を一定数用意する
2. 候補である各視差について、中心の画像におけるウインドウ（5×5）と、各視差に対応する回りの８つの画像におけるウインドウとを比較し、ウインドウ間（８ペア）で値が一致しないピクセルの数 error の合計を計算する
3. error の合計が最小である推定視差を、そのピクセルにおける視差とする

1.　視差の候補について

視差の候補は 0 から 49 までとする。

2.　error の計算について

本プログラムでは、error の計算を以下の様に行う。

①真ん中のピクセルが一致しない推定が正しいとは考え難い。そこで、まず二つのウインドウの真ん中のピクセル同士を比較し、一致しない場合は無条件に error＝25 とする。
②真ん中のピクセルが一致する場合、残りの 24 個の比較をすべて行い、新しく 5×5 のウインドウを作成して、一致、不一致のデータを格納する（一致すれば 0、不一致ならば１を格納）。
③そのウインドウの外周の 0 が、一定数以上の１に囲まれている場合、その 0 を１に書き換える。

（図１において、灰色の部分が外周の 0 にあたる）

ウインドウの外周においては、オクルージョンなどの影響により、注目しているピクセル同士が一致していても、周りに不一致が多い場合はその一致は“偶然の一致”であることが多い。そこでそのような一致を不一致であるとしてデータを書き換えることにより、“偶然の一致”による影響を削減する。例として、ウインドウ間の比較を行って図１が得られた場合、そのデータを図２の様に書き換え、error＝10 が得られることとなる。

0	0	1	0	1
1	1	1	1	1
0	0	0	0	0
0	0	0	0	0
0	0	0	0	0

図１

1	1	1	1	1
1	1	1	1	1
0	0	0	0	0
0	0	0	0	0
0	0	0	0	0

図２

以上のアルゴリズムを用いることで、error を計算する際にウインドウ間で偶然に一致してしまうようなピクセルの数を減少させることができ、視差の推定間違いを減少させることが可能である。

処理結果

以下に PRMU 研究会の行った計測結果を示す。

（実験画像は 9 枚）

Level3(エラー率)		
最小	平均	最大
0.06	2.07	13.98

まとめと考察

error の計算に独自の工夫を加えた結果、精度を上昇させることに成功した。また、凹凸の変化の激しい画像に対してもある程度の精度の上昇がみられ、本手法の効果が確認された。

本プログラムの場合、オクルージョンに対してこれといった工夫が成されていないため、その点を改善することでより高い精度での視差計算が可能である。