コンピュータ理工学部特別研究報告書

題名

立体 360° 画像の自動合成に関する研究

学生証番号　444477

氏名　川西　洋嗣

提出日　平成 30 年 1 月 25 日

指導教員　蚊野　浩

京都産業大学
コンピュータ理工学部

要約

我々の研究室では，4 枚の 360° 画像から立体 360° 画像を生成する研究を進めている．我々のシステムでは，次のように立体 360° 画像を生成する．

(1) 4 枚の 360° 画像を 4 分割し，16 枚の部分画像に分ける．

(2) 16 枚の部分画像から、左目用に 4 枚、右目用に 4 枚の画像を適切に選択し、それらを横に並べることで左目用 360° 画像の初期画像と右目用 360° 画像の初期画像を生成する．これら 2 枚の 360° 画像は，概ね，立体 360° 画像になっているが，画像をつなぎ合わせた境界において，縦方向にズレが生じる．

(3) 縦方向のズレを解消し，滑らかな立体 360° 画像を生成する．

縦方向に生じるズレを解消するために,従来は,人間による手作業で特徴点を対応づけ,幾何学的な変換を行っていた．本研究の目標は，これを自動化することである．

この処理は，画像を貼り合わせる位置において適切な特徴点を抽出する処理と、抽出した特徴点を対応づける処理からなる．第一の処理において，画像を貼り合わせる縦線に沿ってエッジ強度の評価値を計算し，それが閾値よりも大きく，かつ極大になる位置として特徴点を抽出した．第二の処理において，まず，一方の特徴点に対応する位置の候補をテンプレートマッチングで求める．同様に，他方の特徴点に対応する位置の候補をテンプレートマッチングで求める．この結果を照らし合わせ，相互に同じ特徴点にマッチングした場合に，２つの特徴点が対応する特徴点であると推定した．

提案手法により立体 360° 画像の生成を自動化できた．ただし，次の課題が残った．

(1) 特徴点を抽出するための最適な閾値を自動的に決めることができなかった．

(2) 特徴点の対応づけに失敗する場合がある．

目次

1章　序論

コンピュータ内にモデル化した環境を，人間が体験できるようにする技術を仮想現実感（Virtual Reality）とよぶ．仮想現実感の主要な要素の一つに視覚情報があり，そこで利用される画像や映像を VR 画像・VR 映像とよぶ．代表的な VR 画像に立体画像や 360° 画像がある．

両眼立体視に基づく立体画像技術は映画産業や医療分野，ゲーム産業に利用されている．任天堂の 3DS は裸眼で立体視を楽しむことができる．SONY の PlayStation VR は，テレビゲームの映像をヘッドマウントディスプレイ（HMD）に提示することで，迫力のある映像体験を可能にした．

360° 画像は全周囲方向の視野を一枚の画像に記録したものである．頭部の動きなどによって視線方向を指示し，HMD に画像や映像を表示することで没入感がある映像体験を可能にする．YouTube の 360° VR は，全周囲撮影できるカメラ（360° カメラ）で撮影した動画をユーザ同士で共有するサービスである．実写の 360° 動画を撮影するには，RICOH の THETA や SamsungGear360 といった 360° カメラを用いる．

通常の 360° カメラを用いて撮影した一枚の画像は，そのままでは立体視することができない．360° 画像で立体視可能なものは，2 枚の 360° 画像であって，全ての視線方向において適切な両眼視差を有するものである．そのような 2 枚の 360° 画像は，単に，２台の 360° カメラを並置して撮影するだけでは取得することができない．何か特別な工夫が必要である．

我々の研究室では，RICOH の THETA を用いて 4 枚の 360° 画像を撮影し，その 4 枚の 360° 画像から両眼立体視するための 2 枚の 360° 画像を生成するという研究を進めている [1]．この研究ではそれぞれの 360° 画像を縦方向に 4 つに分けることで 16 枚の部分画像に分解する．16 枚の部分画像から左目用の部分画像 4 枚，右目用の部分画像 4 枚を選択する．それぞれ 4 枚の画像を水平方向につなぎ合わせることで左目用の 360° 画像と右目用の 360° 画像を生成する．この時，画像をつなぎ目において縦方向に位置のズレが生じるため，それを解消する必要がある．我々の従来手法では，そこで手作業による修正を行っていた．

本研究では，この研究の撮影方法を利用し，手動で行っていたつなぎ目におけるズレの解消を自動化する手法に取り組む．4 枚の 360° 画像を撮影し，画像を張り合わせて立体 360° 画像を生成する流れがすべて自動化できれば，立体 360° 動画像の撮影が視野に入ってくる．画像を張り合わせるときに発生するずれの確認，対処方法の検証を行う．

2章　立体 360°画像生成をする手法の説明

2 章では 360°画像の生成や撮影に関する従来方法を説明する.

2.1　360°画像の生成

360°画像の撮影には様々な方法がある．例を挙げると，Google がストリートビューの撮影に使っているシステムがある．これは，自動車の上に球型のカメラシステムを載せ（図 2.1），車で少しずつ進みながら 360°画像を撮影する．それを Google マップと組み合わせて,マップで指定した位置の 360°画像が観覧することが可能となっている.

図 2.1　Google ストリートビューで利用された 360°カメラの一例

このような専用のカメラシステムを使用しなくても 360°画像を生成することはできる．普通のカメラの視野方向を地面と平行にし，それを 360°回転させながら撮影した画像を貼り合わせると 360°画像が生成できる．しかしこの方法で生成した場合，天頂の視野が欠損してしまう.この欠損を防ぐには視野方向を平面上だけではなく球面上で移動させながら撮影する必要がある．全周囲方向の視野欠損がない 360°画像を全天球画像，全周パノラマ画像などと呼ぶこともあるが，この論文ではそのような画像もすべて 360°画像と呼ぶことにする.

RICOH の THETA（図 2.2）は代表的な 360°カメラである．表裏に 1 組の魚眼レンズを取り付けており，一度のシャッターで表裏 2 枚の魚眼画像，または，動画を撮影することができる.

一般的なカメラのレンズは画角が約 40～60°なのに対して,魚眼レンズは約 180°と広い画角で撮影することができるので，一度に撮影できる範囲が大幅に広くなる．撮影した 2 枚の魚眼画像に基本的な画像処理（2 枚の明るさ，色彩などの補正）を行い，その後につなぎ目に違和感がないように繋げて，360°画像の生成を行う.

図 2.2　THETA S

2.2　立体 360°画像生成の従来手法

THETA のような 360°カメラで撮影した画像に奥行きの情報はないため，立体視することはできない．両眼立体視を行うには左目用 360°画像，右目用 360°画像をそれぞれ左目，右目に提示する必要がある．この 2 枚の画像をステレオ画像，ステレオ画像を撮影するカメラをステレオカメラ（図 2.3 左）とよぶ．2 台のカメラを人の目幅と同じ間隔に設置し，そのまま 2 台のカメラを同じ間隔のまま回転させながら撮影する．

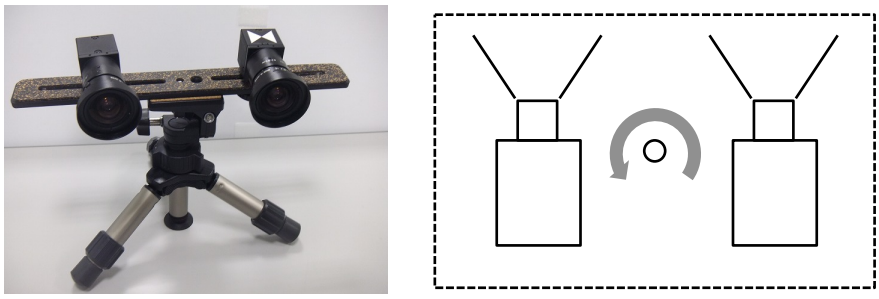


図 2．3　ステレオカメラの例とそれを回転させながら撮影する様子

参考文献［1］より引用

立体 360°画像は図 2.3 の装置を回転ステージに据付け，回転させながら連続撮影した多くの画像から生成することができる．しかしこのステレオカメラを回転させながら撮影する方法では立体 360°の静止画を撮影することは可能だが，動画を撮影することはできない．

2.3　立体 360°画像の撮影，生成方法

THETA でステレオカメラを構成しても，それで撮影した 2 枚の 360°画像だけでは立体 360°画像にはならない．図 2.4 は 2 台の THETA とそれで撮影した 360°画像を模式的に示したものである.説明のための 360°画像を図のように 4 箇所の部分領域に分け，それぞれに L-A，R-A のように名前をつけた．L-A 画像を左目に，R-A 画像を右目に提示すると，正しく両眼立体視ができる．L-B 画像を左目に，R-B 画像を右目に提示すると，正しく立体視ができない．なぜなら L-B 画像と R-B 画像には通常の視差が存在しないからである．L-C 画像を左目に，R-C 画像を右目に提示しても，正しく立体視ができない．この場合は，視差が逆方向に発生しているからである．L-C 画像・R-C 画像を提示する目を入れ替えれば立体視できる.L-D 画像･R-D 画像の組み合わせも立体視ができない．

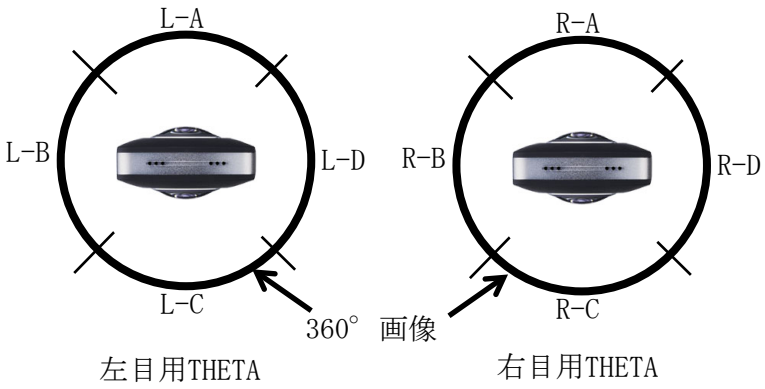


図 2.4　THETA を用いたステレオカメラ

参考文献［1］より引用

立体 360°画像を撮影する従来手法は，図 2.3 のようなステレオカメラを 360°回転させる方法か，Google Jump のように多数のカメラを円形状に配置するカメラシステムを用いる方法である．前者の方法では動画を撮影できない．後者の方法は装置が大規模になってしまう．これらの課題を解決するために，我々は 4 台の 360°カメラを利用する手法を提案している［1].

図 2.5 を用いて我々の手法を説明する.4 台の THETA を円周に 90°の間隔で配置する．４台の THETA で撮影した 360°画像のそれぞれを４分割する．４分割は，(a)のように，隣接するカメラの投影中心を結んだ面と 360°画像の画像面が交差する位置で行う．４分割した部分画像に，L1～L4，R1～R4 の名前を与える．そして，これらを(b)のようにつなぎ合わせる．

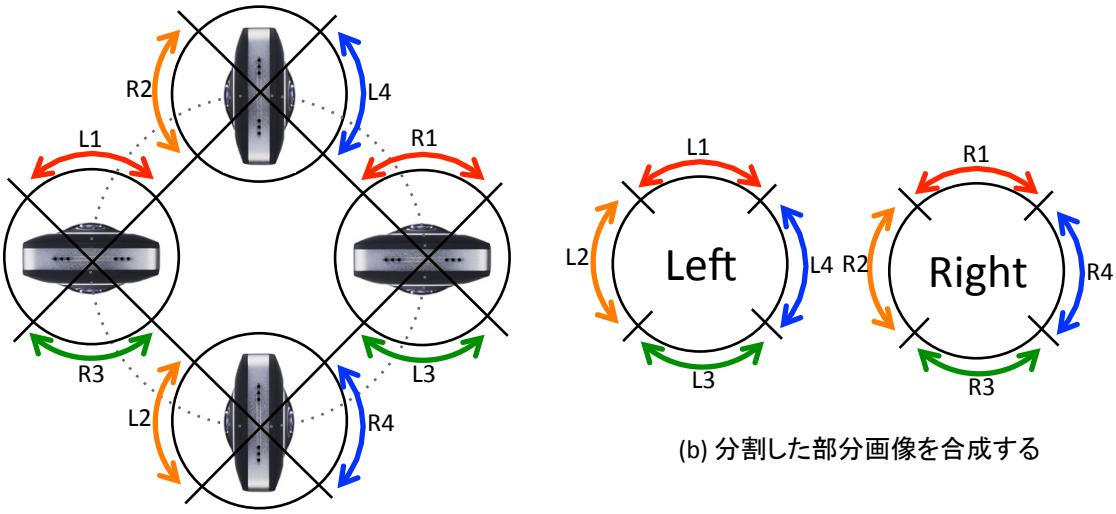


図 2.5　提案手法の概要を説明する図

参考文献［1］より引用

この２枚の画像を立体 360° 画像として提示することを考える．L1 を左目に，R1 を右目の提示した場合，正しく両眼立体視が可能である．なぜなら(a)で確認できるように，L1・R1 は正しいステレオ画像対であるから．同様に L2・R2，L3・R3，L4・R4 の組み合わせも，正しく両眼立体視が可能である．ここまでが提案手法の前半であるが，これだけの処理で，概ね立体 360° 画像が生成できていることがわかる．

図 6 に４箇所で撮影した４枚の 360° 画像の例を示す．これらは，THETA を回転ステージに乗せ，図 2.5(a)のように，90° ごとに撮影した．これらはお互いに 90° に応じた画素数だけ水平方向にずれているが，360° 画像としてはほとんど同じ内容である．ただし，単純な回転移動ではなく位置の移動を伴うため，若干の視差がある．

図 2.6　図 2.5 の 4 か所で撮影した 4 枚の 360° 画像の例

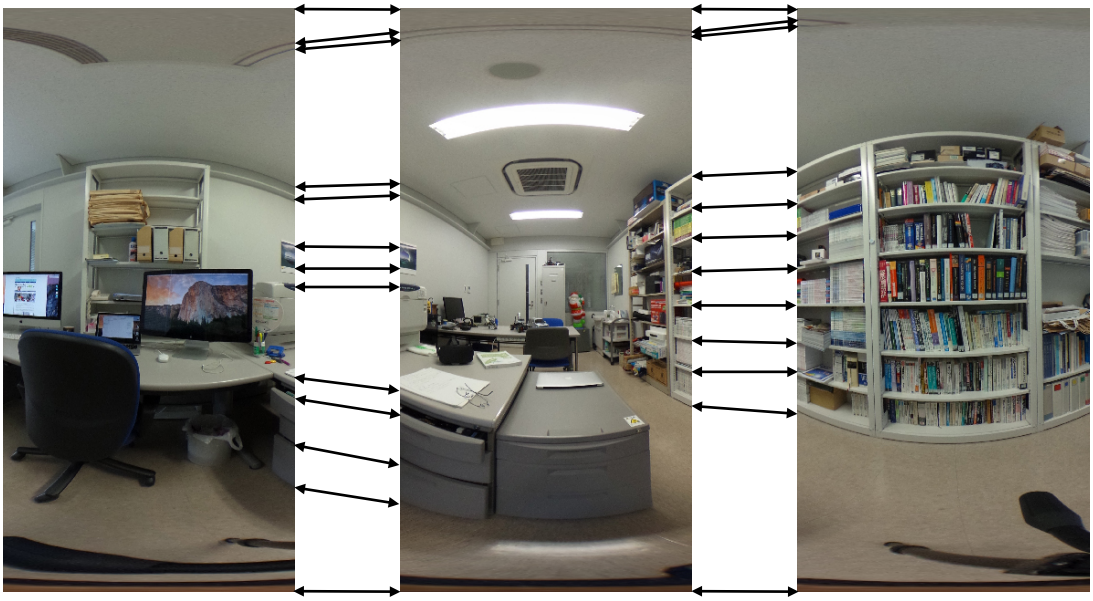
この論文では参考文献［1］と同じ画像を使用する．

これらの画像から，図 2.5 で説明した方法で生成した画像の一つを図 2.7 に示す．図 2.7 の画像は，概ね正しい 360° 画像になっているが，画像のつなぎ目が少し不自然である．よく観察すると，部分画像をつなぎ合わせた位置では，上下方向にずれが生じている．ただし，水平方向のずれはない．これは，360° 画像を 4 分割するときに，隣接するカメラの投影中心を結んだ面で分割したことが理由である．

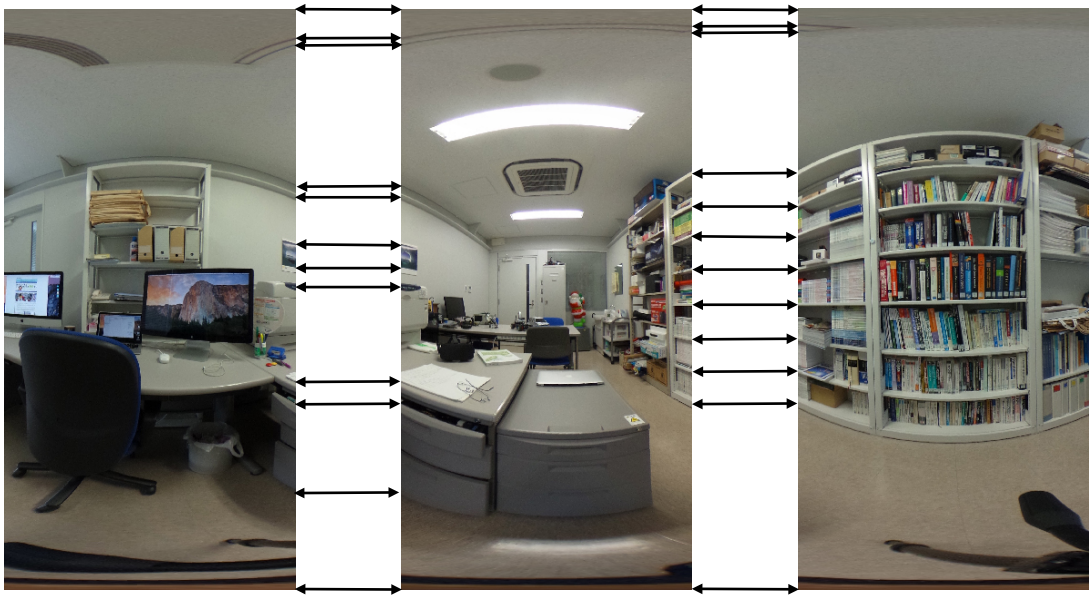
図 2.7　図 2.6 で説明した方法で生成した画像の一方

従って，ここまで説明した方法だけでは望ましい立体 360° 画像を生成することができない．部分画像のつなぎ目が連続するように補正する必要がある．その補正処理の概

要を，図 2.8 を使って説明する．補正処理の第一段階では，隣接する部分画像のずれを対応させる．第二段階では，そのずれが解消するように幾何学的な変換を加える．このような補正処理を加えた部分画像をつなぎ合わせることで，不自然なずれを解消する．



(a) 隣接する部分画像は境界で上下にずれるため，ずれの位置を対応させる．



(b) ずれが解消するように幾何学変換を施し，合成する．

図 2.8　部分画像のずれを補正する処理
参考文献［1］より引用

特徴点を対応付けた後に行う幾何学的な変換を，図 2.9 を用いて説明する．図 2.9 は部分画像 A、B、C をつなぎ合わせる様子である．A と B、B と C の境界で，図のように

対応づけられているものとする．部分画像 B の左端において，対応点の位置が異なる場合，それらの中央位置に移動させる．上端と下端は移動させない．対応点が無い画素の移動量は，その画素を挟む２つの対応点の，それぞれの移動量を線形補間することで求める．部分画像 B の右端についても同様に，全ての画素の移動量を求める．左端と右端以外の画素については，その画素が属する走査線の両端の画素の移動量を線形補間することで求める．このようにして，全ての画素の幾何変換を行う．

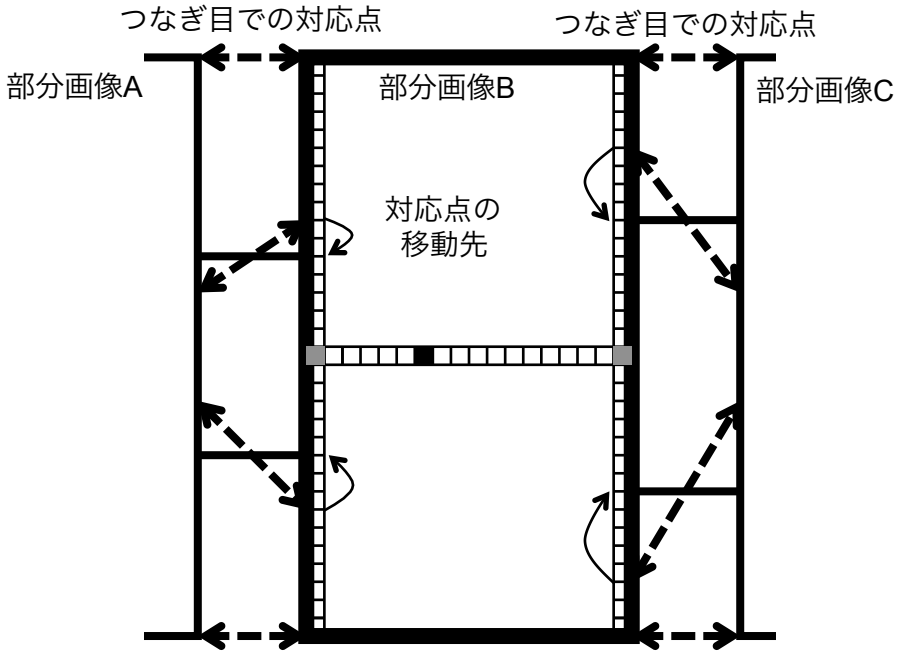


図 2.9　幾何学変換の一般的な方法

3章　立体 360°画像の自動合成の手法

3 章では立体 360°画像を自動的に生成するアルゴリズムを説明をする.

3.1　360°画像の自動合成のアルゴリズム

従来の手法では，貼り合わせる画像の端の特徴となり得る点（以後，特徴点とよぶ）を決めて，その特徴点の y 座標（画像の横方向の位置を x 座標，縦の位置を y 座標とする）を記録し，画像の補正を行っている．人の目で一方の画像から特徴点を決め，それに対応する点（以後，対応点とよぶ）を結びつけ，ずれを修正しる．以上の 360°画像生成のアルゴリズムを図 3.1 に示す. 動作の流れは自動で行うものと手動で行うものと変わりはない．4 分割して使用する画像は図 2.6 と同じである．また，THETA から入力される画像はグレースケール画像で入力し，処理を簡易化する.

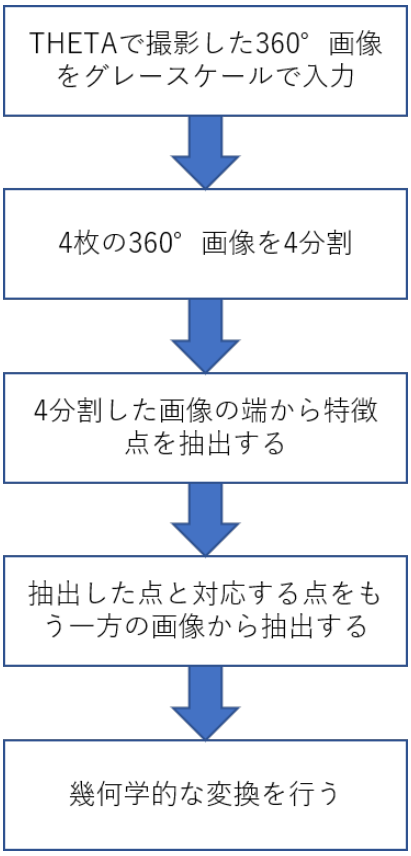


図 3.1　立体 360°画像生成のアルゴリズム

3.2　4 分割した画像の端から特徴点を抽出する.

特徴点の抽出方法の説明をする. 特徴抽出の既存技術に, SIFT, それを簡易化し高速化した SURF, AKAZE といった様々な技術がある. しかし, これらの技術は角 (コーナー) を特徴としてとらえる技術のため, 画像端から特徴を抽出することは分割した画像端にコーナーがない限り特徴点としてとらえてくれない. よって, この研究ではコーナーでの特徴抽出は行わず, エッジを特徴として抽出する.

今回の研究で最終的に表示する画像は図 3.2 の赤枠に囲まれた部分である.

図 3.2　撮影した 4 枚の 360° 画像

(赤枠線で囲まれた範囲が合成後に表示される画像)

図 3.2 の 4 枚の画像を貼り合わせる際に必要な特徴点を抽出するのだが, どのような場所を特徴として抽出するかを図 3.3 の画像を使用して説明する. 特徴点を取り出す場所の条件として, 図 3.2 に描かれている赤枠縦線上にある特徴のみを利用する. これは, 特徴点を対応させる処理を行うとき, 対応する場所を探索する範囲が広がり, 処理時間が長くなる他, 横方向の情報も加わるため誤って対応させてしまう可能性が高くなってしまうなどの理由がある. どのような場所が特徴点となるか示した例が図 3.3 である. 図 3.3 にある青色の矢印が示している場所は, 人間が特徴点として認識する場所である. 青色の矢印は, 赤枠と横方向のエッジが交わった場所を示している. このようなエッジは特徴としてわかりやすいため, このような場所を検出する.

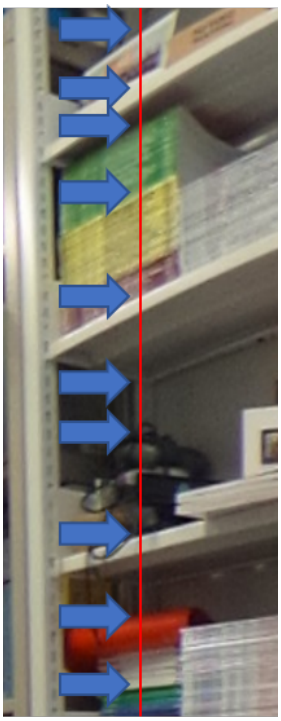
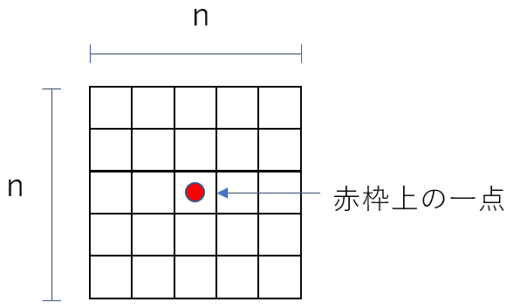
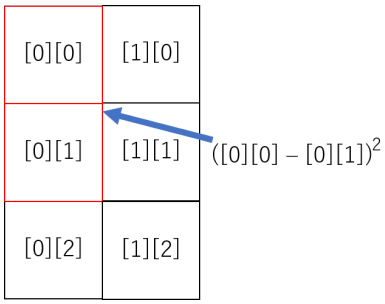


図 3.3　図 3.2 の右上の画像の赤枠の右側あたりを拡大した画像
その画像の中で特徴点となり得る場所を矢印で示した

次に特徴点の抽出方法を説明する．横方向のエッジ近傍において，画素値は縦方向に大きく変化する．このため，縦方向の画素値の差分は他の場所と比べ大きな値をとる．この性質を利用した特徴抽出の方法を表したものが図 3.4 である．(1)まず特徴点が存在する赤線上の点を中心とした n×n 画素の範囲をとり，(2)その範囲内で上下に隣接する画素値の差分を求める．その差分は負の値になることもあるので，二乗してすべて正の値にし，(3)二乗した値を合計した数値を特徴量として記録する（この数値を今後は特徴量とよぶ）．横方向のエッジがある場所でこの計算をすると，特徴量は大きな値を取る．適当な閾値を設定し画素値の変化が大きい点だけを抽出することで赤枠上のエッジを抽出することができる．



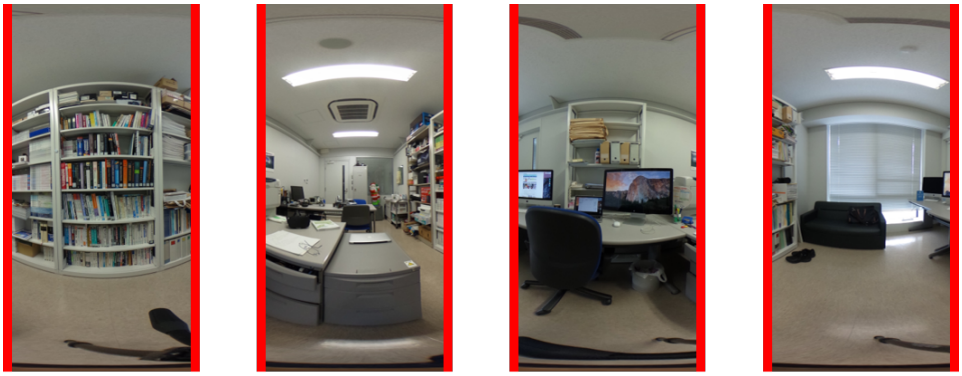
(1) 赤枠線上の一点を中心とした n×n 画素の範囲を取る．



(2) 上下に隣接している画素値の差分を取り二乗する．

$$([0][0] - [0][1])^2 + ([0][1] - [0][2])^2 + \cdots + (([n-1][n-2]) - ([n-1][n-1]))^2$$

(3) (2)の処理を範囲内で繰り返し，合計する．

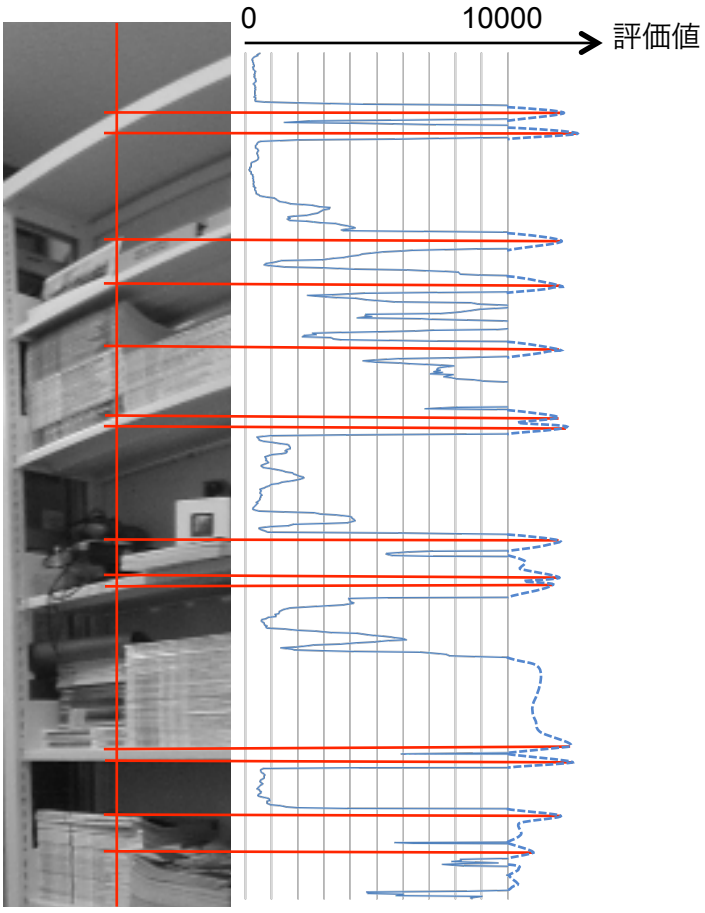


(1)から(3)の処理を赤枠縦線上の全画素で行う．

図 3.4　特徴量の計算

以上の動作をウィンドウサイズ 11×11 で確認した結果を図 3.5 に示す．明確なエッジが存在する位置では特徴量の値は 10000 を遥かに超える．この図から，特徴量が閾値以上であり，局所的に最大である位置を特徴点として抽出すれば良いことがわかる．本研究では閾値を 2500 とし，閾値を超える区間において極大となる点を特徴点とした．

実験後に気がついたことであるが，評価値が極大となる位置の中で，その極大値が閾値よりも大きいものを特徴点とする方が良かったと思われる．



ウィンドウサイズは 11×11

図 3.5　差分の二乗和とエッジ特徴の関係

3.3　抽出した特徴点と対応する点の探索

画像境界で検出した特徴点を対応付ける方法を，図 3.6 を用いて説明する．まず，一方の画像から抽出した特徴点に対応する他方の画像の位置の候補をテンプレートマッチングで探索する．同様に他方の画像から抽出した特徴点に対応する一方の画像の位置の候補をテンプレートマッチングで探索する．このように双方向で探索した結果を照合し，両者の特徴点が同じ位置でマッチングした場合に，対応付けに成功したとする．

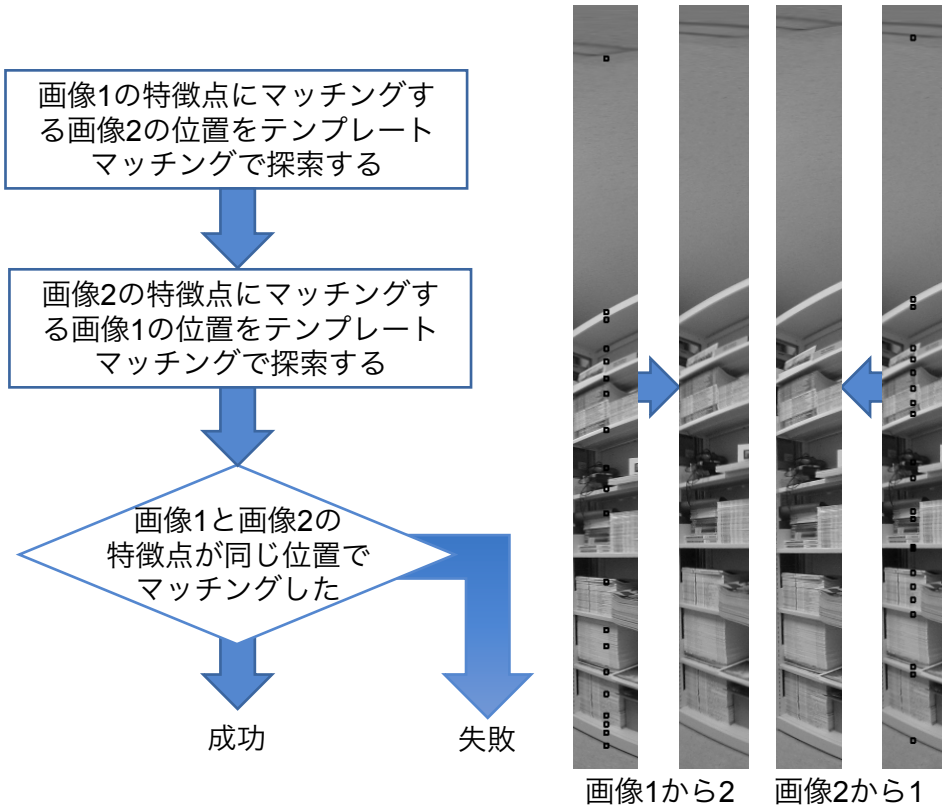


図 3.6 特徴点を対応づける方法

テンプレートマッチングに用いるウィンドウサイズと類似度の評価法を変えて，対応付けの精度を確認した．ウィンドウの大きさは 11×11 画素，21×21 画素，25×25 画素，31×31 画素で比較した．類似度は正規化相互相関係数（NCC，Normalized Cross Correlation）と SSD（Sum of Squared Difference）の 2 通りで比較した．また，テンプレートマッチングによる一方向からの対応付けと双方向から対応付けを比較した．実験結果を表 3.1(1)から(4)に示す．

図 3.1(1)はウィンドウサイズ 11×11 の場合の結果である．「画像 1 から 2」で SSD の欄に 18/22 とあるのは、抽出した特徴点が 22 個あり，それをウィンドウサイズ 11×11 の SSD でテンプレートマッチングさせた．その結果を目視で確認したところ正解が 18 個であった、という意味である．「双方向」で SSD の欄に 13/14 とあるのは，テンプレートマッチングの結果を双方向からプログラムで検証した結果，14 個を対応付けに成功と判断した．それを目視で確認したところ 13 個が正解であった，という意味である．

表 3.1(1) ウィンドウのサイズ：11×11

	画像 1 から 2	画像 2 から 1	双方向
SSD	18/22	19/22	13/14
NCC	16/22	18/22	11/12
	画像 2 から 3	画像 3 から 2	双方向
SSD	19/22	18/24	10/11
NCC	17/22	18/24	10/11
	画像 3 から 4	画像 4 から 3	双方向
SSD	21/23	22/24	17/17
NCC	19/23	22/24	17/17
	画像 4 から 1	画像 1 から 4	双方向
SSD	12/15	8/11	7/7
NCC	10/15	7/11	5/5
合計			90/94

表 3.1(2) ウィンドウのサイズ：21×21

	画像 1 から 2	画像 2 から 3	双方向
SSD	20/22	21/22	15/15
NCC	20/22	20/22	15/16
	画像 2 から 3	画像 3 から 2	双方向
SSD	21/22	18/24	11/11
NCC	19/22	20/24	11/11
	画像 3 から 4	画像 4 から 3	双方向
SSD	23/23	22/24	18/18
NCC	20/23	22/24	17/18
	画像 4 から 1	画像 1 から 4	双方向
SSD	12/15	10/11	8/8
NCC	11/15	10/11	7/7
合計			102/104

表 3.1(3) ウィンドウのサイズ：25×25

	画像 1 から 2	画像 2 から 1	双方向
SSD	20/22	21/22	14/14
NCC	21/22	22/22	15/15
	画像 2 から 3	画像 3 から 2	双方向
SSD	20/22	21/24	11/11
NCC	19/22	20/24	11/11
	画像 3 から 4	画像 4 から 3	双方向
SSD	23/23	22/24	18/18
NCC	21/23	22/24	18/19
	画像 4 から 1	画像 1 から 4	双方向
SSD	12/15	9/11	7/7
NCC	11/15	9/11	7/7
合計			101/102

表 3.1(4) ウィンドウのサイズ：31×31

	画像 1 から 2	画像 2 から 1	双方向
SSD	22/22	22/22	15/15
NCC	22/22	22/22	15/15
	画像 2 から 3	画像 3 から 2	双方向
SSD	21/22	22/24	11/11
NCC	20/22	21/24	10/10
	画像 3 から 4	画像 4 から 3	双方向
SSD	23/23	23/24	19/19
NCC	23/23	22/24	18/18
	画像 4 から 1	画像 1 から 4	双方向
SSD	14/15	11/11	9/9
NCC	14/15	11/11	8/8
合計			105/105

表 3.1 全体を見ると，マッチングウィンドウが大きいほど誤対応が少ない．また，双方向から探索した時に，対応付けできる点の数と成功する割合も増えた．31×31 以上

マッチングウィンドウを大きくすると使用メモリが大幅に増えてしまうため，31×31で実験を続行する．

4章　提案手法の結果，考察

4.1　生成画像の例

図 4.1 に幾何学的変換を行う前の画像を示す．つなぎ目にずれが生じている．図 4.2 が修正後の最終画像である．

図 4.1　図 2.6 で説明した方法で生成した画像

図 4.2　自動合成した 360° 画像

図 4.1 と図 4.2 を比べると，図 4.1 よりも図 4.2 の方がつなぎ目が滑らかになっている．しかし，まだずれが完璧に直せたわけではなく，画像上端の方や，細かいところでまだずれが生じている場所が確認できる．

4.2　考察

画像のつなぎ目の修正と自動合成を実装することができた．しかし今後の課題として次があげられる．

（1） 特徴抽出が完全ではなく，人の目で見ると明らかに特徴となり得る場所も抽出できていない．

（2） 不完全なずれの修復．ずれが残っている場所や，拡大してみるとごくわずかにずれが残っていて，手動のものと比較すると明らかに自動合成の方が精度は悪い．原因はまだ誤対応が残っているため．

（3） 横方向への対策．本来，横方向にずれが生じることはないが，人の手で撮影したこともあり，ごくわずかに横にずれてしまっている．テンプレートマッチングをする際に，この横方向のずれが誤対応につながってくる．

5章　結論

本研究ではまだ精度がいいわけではないもののずれの修正を行うことができた.特徴点の抽出や対応点問題を解決するとTHETAで立体360°動画像の撮影をするようになれるのも近いうちに実現できるようになるはずである.

また，作成したプログラムは処理にある程度時間がかかり，撮影から立体360°画像を表示するまでに処理が追い付かない可能性もある.また，この研究ではエッジを特徴として抽出していたが,エッジを特徴とするよりももっと効率よく処理できるアルゴリズムも研究すればアルゴリズムの改善も今後の課題となる.

参考文献

［1］　高木 亮佑,「４枚の 360° 画像のつなぎ合わせによる立体 360° 画像の生成に関する研究」, 京都産業大学 コンピュータ理工学部 特別研究 II, 2017 年 1 月.

謝辞

本論文を作成にあたり，丁寧なご指導を賜りました蚊野浩教授に感謝いたします.