動的計画法に基づく単調連続
2 次元ワープ法に関する研究

内田誠一

1999 年 1 月

目 次

1 序論 **4**

1.1 2 次元ワープの概要 4

1.2 本研究の方針 5

1.3 論文の構成 6

1.4 従来の研究 7

1.5 動的計画法に基づく 2 次元ワープ法の分類 10

2 単調連続 2 次元ワープ決定問題 **12**

2.1 2 次元ワープ決定問題の定式化 12

2.2 単調連続性条件と境界条件 13

3 動的計画法 **15**

3.1 動的計画法の概要 15

3.2 多段決定過程 17

3.3 ビームサーチ 19

4 動的計画法による解法 1 – カラムワイズ DP アルゴリズム **21**

4.1 まえがき 21

4.2 列を処理単位とする多段決定過程による問題表現 21

4.3 カラムワイズ DP アルゴリズム 23

4.4 計算量の評価 25

4.5 近似アルゴリズム – カラムワイズビームサーチ DP アルゴリズム . . 26

4.6 計算量の評価 30
4.7 まとめ 31

5 動的計画法による解法 2 – ピクセルワイズ DP アルゴリズム 32
5.1 まえがき 32
5.2 画素を処理単位とする多段決定過程による問題表現 32
5.3 ピクセルワイズ DP アルゴリズム 37
5.4 計算量の評価 38
5.5 近似アルゴリズム – ピクセルワイズビームサーチ DP アルゴリズム . 39
5.6 計算量の評価 43
5.7 まとめ 44

6 アルゴリズムの比較 46
6.1 まえがき 46
6.2 DP アルゴリズムの比較 46
6.3 ビームサーチ DP アルゴリズムの比較 47
6.4 まとめ 52

7 不自然なワープの回避 53
7.1 まえがき 53
7.2 ペナルティ 53
7.3 整合窓 57
7.4 まとめ 58

8 評価実験 59
8.1 まえがき 59
8.2 単調連続 2 次元ワープの結果例とその定性的評価 59
8.3 変形追従能力の定量的評価 67
8.4 手書きひらがな認識による評価 71
8.5 まとめ 75

9 結論 **77**

謝辞 **79**

参考文献 **79**

A DP 漸化式の導出 **87**

A.1 **カラムワイズ DP アルゴリズムが基づく DP 漸化式の導出** 88

A.2 **ピクセルワイズ DP アルゴリズムが基づく DP 漸化式の導出** 90

第 1 章

序論

1.1　2 次元ワープの概要

本論文では，2 画像の最大一致を与える画素間のノンパラメトリックなマッピングを 2 次元ワープと呼ぶ[1]．2 次元ワープは画像中のパターンの変形に適応可能な弾性マッチング処理と見なすことができる．また一方の画像を，画素をプリミティブとするモデルと考えると，構造解析処理にもなっている．

2 次元ワープを決定する問題は一種の最適化問題であり，その定式化には二通りのアプローチがある．第一は，画像を画素の集合と見なし，2 集合の要素間の最適対応関係を求める組合せ最適化問題 [2], [3] として定式化するものである．多くの組合せ最適化問題は組合せ爆発の問題を抱えているので [4]，アルゴリズムの検討にあたっては計算量的考察が重要となる．また場合によっては近似アルゴリズムの検討も必要となる．第二は，画像を 2 次元連続信号と見なし，汎関数を極小 (大) 化する変関数を求める変分問題として定式化するものである．この場合，R^2から R^2への連続関数として表現される 2 次元ワープが変関数であり，画像の一致度を評価する目的関数が汎関数である．計算機上では，極値の必要条件であるオイラー・ラグランジュ方程式を離散近似したものを反復法により解くことになる．誤差の発生要因が多いため，解の安定性を保つための配慮が必要となる．

2 次元ワープの性質は，ワープによる変形の局所的自由度と，最適性の有無によって特徴付けられる．2 次元ワープの自由度は，ワープに対する制約条件によって規定される．このため制約条件は対象画像の変形特性を反映していることが望ましい．例えば対象の変形がいわゆるゴム膜変形で近似できる場合，位相構造を保存する制約条件が適している．一方，ワープの最適性の有無はワープの変形適応能力

[1] 事前に与えられた幾つかの画素対応関係を内挿する関数も 2 次元ワープと呼ばれている [1]．本論文で扱う 2 次元ワープとは，そのような事前知識は用いずにすべての画素対応を自動決定するものを指す．

の安定性を左右する．すなわち，最適性が保証されていれば，制約条件を満たす範囲内で常に最大限の変形適応能力が発揮される．ワープの最適性の有無は前述の定式化と関連し，そこで用いる最適化手法によって規定される．

1.2 本研究の方針

本研究は，以下の基本方針に則しながら，適正な自由度を有し，かつ最適性が保証される 2 次元ワープアルゴリズムの実現，および実用を目指したアルゴリズムの改良を目的として行なったものである．

(1) 位相構造を保存する一般的なワープモデルの定式化: 画像の位相構造を保存する範囲での最大の自由度を考え，それに応じた制約条件を使用する． 多くの場合，対象に非単調な変形や不連続な変形が生じないことを考えると，これは妥当な方針と言える．対象固有の変形特性を詳細に反映した制約条件の利用は先の問題とする．

(2) 最適ワープを求める DP アルゴリズムの確立: 最適性が保証されたワープを与えるアルゴリズムの実現は，理論的に興味深いだけでなく，高精度なワープを志向する意味で実用的にも意義がある．前述のように 2 次元ワープ決定問題には組合わせ最適化問題と変分問題の二つの見方があるが，後者では一般に局所解しか求まらないので，前者の枠組での動的計画法 (dynamic programming : DP) [5]∼[7] に基づくアルゴリズムを設計することとする．

(3)2 次元的自由度を堅持した上での効率化: 前項 (2) に従って 2 次元ワープ決定問題を組合わせ最適化問題として扱う場合，前述のように組合せ爆発の問題が危惧される．実用を目指した計算量の低減には，ワープの自由度に対する制約を強めることで探索空間を圧縮する方向と，最適ワープが得られる可能性の低い探索を途中で打ち切る方向が考えられる．本研究ではワープの自由度に直接影響を与えない後者を基本とする．

1.3 論文の構成

本論文は，前節の方針に従い，単調連続性を制約条件とする 2 次元ワープについて DP に基づくアルゴリズムを提案し，その基礎的な検討を行なったもので，9 章より構成される．

本章の残りでは，2 次元ワープに関する従来の研究を概観する．特に DP に基づくアルゴリズムについては，ワープに対する制約条件とワープの最適性の有無を基準とした分類を試み，それらの問題点を明らかにする．

第 2章では単調連続性を制約条件とする 2 次元ワープの決定問題の定式化を行なう．単調連続性とは隣接画素間の上下左右関係と近傍関係をワープ後も保存するための条件であり，これにより画像の位相構造を保存する 2 次元ワープが実現される．

第 3章では以後の議論の準備として DP の概要を述べる．DP と密接な関係のある多段決定過程と，DP アルゴリズムの計算量低減手法であるビームサーチについても言及する．

第 4章では単調連続 2 次元ワープ決定問題の最適解を与えるアルゴリズムとして，カラムワイズ DP アルゴリズムを提案する．画像サイズを $N \times N$ とすると，このアルゴリズムは画像の列を処理単位とする N 段の多段決定過程に基づいている．各列のワープに単調連続性を保証し，かつ列間の遷移に単調連続性を保証するため，段あたり $2N$ の指数オーダーの探索幅を持つ計算量的に困難な問題となる．この問題に対して，DP アルゴリズムにビームサーチを適用した近似アルゴリズムを提案する．ビームサーチを DP アルゴリズムに組み込むことで最適性の保証はなくなるが，ワープの 2 次元的自由度を保ったまま計算量を多項式オーダーに低減できる．

第 5章では最適ワープを与えるもう一つのアルゴリズムとして，ピクセルワイズ DP アルゴリズムを提案する．このアルゴリズムは各画素のワープをラスタスキャン順に決定していく N^2 段の多段決定過程に基づいている．各段での決定は 2 次元的単調連続性制約により N 段前の決定に依存する．ピクセルワイズ DP アルゴリズムはカラムワイズ DP アルゴリズムを計算量の点で改良したものであるが，この N 次マルコフ性を扱っているために段あたり N の指数オーダーの探索幅を持つ．この

ため，前章と同様にビームサーチを適用した近似アルゴリズムを提案する．

第 6 章ではまずカラムワイズ DP アルゴリズムとピクセルワイズ DP アルゴリズムの計算量の比較を行なう．次にそれぞれの近似アルゴリズムに関しても計算効率の実験的および理論的な比較を行う．

第 7章では，単調連続性の局所性やビームサーチの副作用に起因した不自然なワープを回避する方法として，ペナルティと整合窓の適用を検討する．

第 8章ではピクセルワイズビームサーチ DP アルゴリズムを用いた幾つかの実験により単調連続 2 次元ワープの精度の定性的および定量的な評価を行なう．

最後に第 9章では論文のまとめと今後の課題について述べる．

1.4 従来の研究

画像の弾性マッチングに関する最初の試みは B.Widrow によるラバーマスク法 [8] と言われる．ラバーマスク法では一方の画像を幾つかのパラメータを持つ関数により変形し，最大一致が得られるまでパラメータを繰り返し修正する．このようなパラメトリックな弾性マッチングを決定する問題は目的関数の極値問題として最小 2 乗法や勾配法などにより解くことができる．若原による LAT [9] では，ワープ全体を部分的なアフィン変換に分解して捉え，各領域において最適なアフィン変換のパラメータを最小 2 乗法により決定する．A.K.Jain らにより提案された deformable templates [10] では，調波構造を持つ正弦関数の重ね合わせによってワープを表現しており，各正弦関数の振幅パラメータを勾配法により求めている．他のパラメトリックな弾性マッチング法として，アフィン変換のパラメータを少しずつ変化させて生成した変形画像を予め数多く用意しておく方法 [11]∼[13] があり，文字認識に応用されている．

ノンパラメトリックな弾性マッチング，すなわち 2 次元ワープに関しても多くの研究がなされている．最適化手法をキーとして分類すれば，それらは決定論的弛緩法 [14], [15] などの反復法に基づく方法 [16]∼[21]，摂動法に基づく方法 [22]∼[26]，および DP に基づく方法 [27]∼[38] の 3 種類に大別される．

決定論的弛緩法に基づく方法 [16]∼[20] では変分法の枠組で 2 次元ワープを求める．計算機上では，オイラー ・ラグランジュ方程式の差分近似により構成される大型の非線形連立方程式を Gauss-Seidel 法などの反復法により解くことになる．過変形を回避するために，正則化や多重スケールを用いた階層的な解法が用いられる．これらの方法の欠点は，オイラー ・ラグランジュ方程式が最適解の必要条件であるために，たとえ反復計算が収束しても一般に局所解しか得られない点である．またワープが実数値関数として扱われるために，計算途中で整数座標以外の点の輝度値を参照しなくてはならない点も問題である．決定論的弛緩法以外の反復法に基づく方法として，弾性膜のポテンシャル場への落ち込みを運動方程式で表現し，系の平衡状態を反復法により求めるワープ法 [21] が提案されているが，これも同様の欠点を持つ．

摂動法 (ずらし) に基づく方法 [22]∼[26] では，各画素 (もしくは部分領域) 毎に最も一致する画素を独立に探索することでワープを決定する．これらの方法は最適性を度外視した代償として計算量が他の組合わせ最適化手法よりも格段に少なくて済むという特長がある．しかし探索の独立性と局所性のために，多重スケールを用いた階層的な探索法を用いたとしても，過変形が生じる可能性が高く，加えて大きな変形にも対応できないという欠点を持つ．また同じ理由により制約条件や正則化手法が導入できないこともこの欠点の克服を困難なものにしている．

DP は変分問題の数値解法であると同時に，組合せ最適化問題の代表的な解法でもある．DP は，1) 解の最適性が保証される，2) 目的関数の微分可能性の制約が無い，3) 様々な制約条件を取り扱える，4) 演算誤差の集積による不安定性が無い，などの特長を持つ．DP に基づく 2 次元ワープ法は，音声認識における時間歪み整合 (1 次元ワープ) 問題の DP による解決 [39], [40] に刺激を受け，これまでにも幾つか検討されている． しかし本来 1 次元的な問題を扱う DP を 2 次元な問題を扱えるように拡張するのは容易ではないため，いずれも自由度もしくは最適性を犠牲にしたり，あるいは問題の本質を見失った方法になっており，**1.2**の方針 (1)(2) を同時に満たす方法は知られていない．

DP に基づく従来の 2 次元ワープ法の例として，画像の縦横の周辺分布について

独立にそれぞれ 1 次元 DP を施す方法 [27]，縦方向を固定して横方向に 1 次元 DP を施す方法 [28]∼[30]，縦方向の DP の内部で横方向の DP を行なう方法 [31]∼[36] が挙げられる．これらはいずれも特定方向の伸縮は吸収可能であるが，傾きや回転には対応できず，その意味では完全な 2 次元的自由度を持たない．このため，ステレオマッチング [30] やスペクトログラムのマッチング [31], [32] などの特殊な変形を扱う場合を除いて，画像マッチング問題への有効性はあまり期待できないと考えられる．

2 次元的な自由度を持つ 2 次元ワープ法としては，杉村らの方法 [37] がある．この方法では画像の位相構造を保存するワープを実現しているが，ある列の DP による対応付けの結果を固定して次の列の対応付けの制約とするために，行方向に関する最適性が無い．このため画像の周辺部に複雑なパターンや雑音が存在する場合には特に不利である．一方，磯道ら [38] は，まず縦方向を固定し各行について 1 次元 DP を施し，その結果得られた変形画像に対し今度は横方向を固定して各列について 1 次元 DP を施すという，2 段階のアルゴリズムを提案している．この方法もワープの最適性は保証されない．

Levin ら [33] は，2 次元的な自由度を持ち，かつ画像全体で最適性が保証される 2 次元ワープ法を提案した．しかし用いている制約条件が単調性だけであるために，過変形と計算量の問題が深刻であり，文献 [33] ではアルゴリズムの記述に終わっている[2]．

本論文が扱う 2 次元ワープ決定問題の枠組には直接当てはまらないが，同じく DP を用いた画像マッチング法として，対角方向に対応済みの矩形領域を拡張していく方法 [41]∼[45] がある．これらの方法は Levenshtein 距離の 2 次元的な拡張として提案されたものであるが，文献 [37], [46] でも指摘されているように，画像マッチング法としては不自然な定式化がなされており，有効な応用は見当たらない．

[2]Levin らの方法については，**6.2**で計算量を考察し，**8.2.2**でそのワープによる過変形の実例を示す．

1.5　動的計画法に基づく 2 次元ワープ法の分類

前節で述べた DP に基づく従来の 2 次元ワープ法 [27]∼[38] を，2 次元的自由度を基準として分類した結果を表 1.1に示す．表中の制約条件 (i)∼(iv) は，いずれも 4 隣接する 2 画素のワープ後の位置関係に対する制約を表している．その項目に “≡” や “=” があるものは 2 次元的な自由度が無いことを意味し，“∅” があるものはその方向に関する過変形が危惧される．表中の最適性は，この制約条件の下で最適なワープが求まるアルゴリズムとなっている場合を “○ ” とし，そうでなければ “× ” としている．

各クラス (a)∼(f) に属する 2 次元ワープの例を正方格子 (5×5) の変形として図 1.1 に示す．なお同図の各例では，範囲制約 (“>”) として，(i)(iv) の場合±1 画素，(ii)(iii) の場合 $0 \sim +2$ 画素が便宜上仮定されている．

表 1.1: DP に基づく 2 次元ワープ法の 2 次元的自由度による分類

クラス	制約条件				文献	応用例	最適性	時間計算量
	(i)	(ii)	(iii)	(iv)				
(a)	$=$	$>$	$>$	$=$	[27]	印刷漢字認識	×	$O(N^2)$
(b)	$=$	$>$	$\equiv$	$\emptyset$	[28], [29]	手書き漢字認識	○	$O(N^3)$
(c)	$=$	$>$	$\equiv$	$>$	[30]	ステレオマッチング	×	$O(N^2)$
(d)	$=$	$>$	$>$	$\emptyset$	[31], [32]	不特定話者単語音声 認識 (frequency warp)	○	$O(N^4)$
					[33]∼[36]	手書き英文字 (列) 認識	○	$O(N^4)$
(e)	$>$	$>$	$>$	$>$	[37]		×	$O(N^3)$
					[38]		×	$O(N^2)$
					本論文		○	$O(N^2 9^{2N})$
					本論文		○	$O(N^3 9^N)$
(f)	$\emptyset$	$>$	$>$	$\emptyset$	[33]		○	$O(N^{4N})$

制約条件 : ワープ前に 4 隣接関係にある画素において
(i) 横に隣接する 2 画素のワープ後の縦方向の位置関係の制約
(ii) 横に隣接する 2 画素のワープ後の横方向の位置関係の制約
(iii) 縦に隣接する 2 画素のワープ後の縦方向の位置関係の制約
(iv) 縦に隣接する 2 画素のワープ後の横方向の位置関係の制約
$\equiv$: 絶対位置の維持，$=$: 前関係の維持， $>$: 範囲制約，$\emptyset$: 無制約

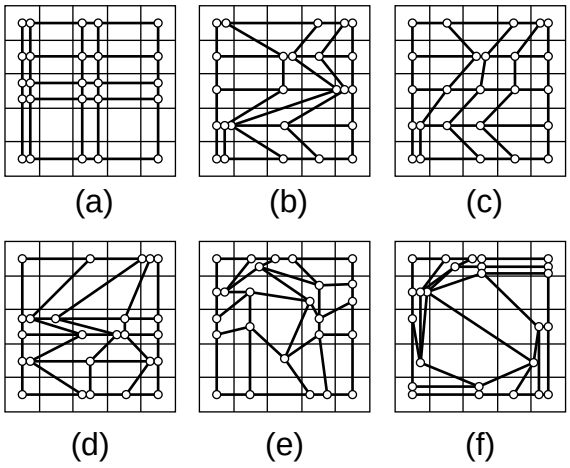


図 1.1: 各クラスの 2 次元ワープの例

第 2 章

単調連続 2 次元ワープ決定問題

2.1　2 次元ワープ決定問題の定式化

ディジタル画像 $\boldsymbol{A} = \{a(i,j) \mid i,j = 1,2,\ldots,N\}$ および$\boldsymbol{B} = \{b(x,y) \mid x,y = 1,2,\ldots,M\}$ を考える[1]．ここで $a(i,j)$，$b(x,y)$ はそれぞれ画素 (i,j)，(x,y) の特徴量とする．画像 $\boldsymbol{A}$ から $\boldsymbol{B}$ への画素間のマッピングを表す関数

$$\begin{aligned} x(i,j) &: \{1,2,\ldots,N\} \times \{1,2,\ldots,N\} \to \{1,2,\ldots,M\} \\ y(i,j) &: \{1,2,\ldots,N\} \times \{1,2,\ldots,N\} \to \{1,2,\ldots,M\} \end{aligned}$$

をワープ関数と呼ぶ (図 2.1) ．このとき 2 次元ワープ決定問題は，何らかの制約条件下で目的関数

$$D = \sum_{i=1}^{N}\sum_{j=1}^{N} \mid a(i,j) - b(x(i,j),y(i,j)) \mid \tag{2.1}$$

の最小値を与えるワープ関数の決定問題として定式化される．このように 2 次元ワープ決定問題はワープ関数を変関数とする変分問題であり，また画素対応関係の組合せ最適化問題にもなっている．なお，このワープ関数により変形された画像 $\{b(x(i,j),y(i,j)) \mid i,j = 1,2,\ldots,N\}$ を $\widetilde{\boldsymbol{B}}$ と表す．

目的関数 D の最小値を $D(\boldsymbol{A},\boldsymbol{B})$ と表す．

$$D(\boldsymbol{A},\boldsymbol{B}) = \min_{\{x(i,j),y(i,j) \mid i,j=1,\ldots,N\}} \sum_{i=1}^{N}\sum_{j=1}^{N} \mid a(i,j) - b(x(i,j),y(i,j))\mid \tag{2.2}$$

パターンマッチングの立場から見ると，$D(\boldsymbol{A},\boldsymbol{B})$ は，2 次元ワープによって変形された画像$\boldsymbol{B}$のうちで$\boldsymbol{A}$に最も近いものと，画像$\boldsymbol{A}$との距離に相当する．また，構造解析の立場から見れば，この時のワープ関数 $(x(i,j),y(i,j))$ は，モデル$\boldsymbol{B}$による$\boldsymbol{A}$の一つの解釈を与えている．

[1] 以後の議論を縦横の長さの異なる画像に一般化することは容易である．

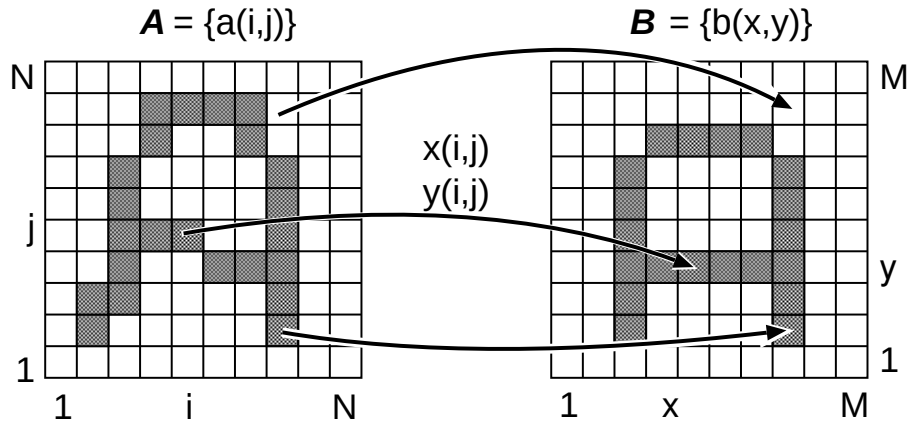


図 2.1: 2 次元ワープ

2.2 単調連続性条件と境界条件

ワープ関数に対する制約として 2 次元的単調連続性条件を用いる．ここでは単調性および連続性を，それぞれ隣接画素の上下左右関係および近傍関係がワープ後も保存されることと定義する (図 2.2(a)(b))．この単調連続性条件は次式で与えられる (図 2.2(c))．

$$0 \leq x(i,j) - x(i-1,j) \leq 2 \tag{2.3}$$

$$0 \leq y(i,j) - y(i,j-1) \leq 2 \tag{2.4}$$

$$|x(i,j) - x(i,j-1)| \leq 1 \tag{2.5}$$

$$|y(i,j) - y(i-1,j)| \leq 1 \tag{2.6}$$

単調連続性の効果を図 2.3の画像を例として説明する．多くの場合 $\boldsymbol{B}_1 \sim \boldsymbol{B}_3$ のうちで$\boldsymbol{A}$ との距離を最も小さくしたいものは $\boldsymbol{A}$ と同相な $\boldsymbol{B}_3$ であると考えられる．しかし，連続性が単独で用いられると，$\boldsymbol{A}$ が部分的に反転したパターンである $\boldsymbol{B}_1$ との距離も 0 に近い値となる． また単調性が単独で用いられると，不連続に変形されている $\boldsymbol{B}_2$ との距離が 0 に近い値となる[2]．このように，いずれの場合も過変形の

[2] **8.2.2**では単調性のみを制約とする 2 次元ワープによる過変形の例を示す．

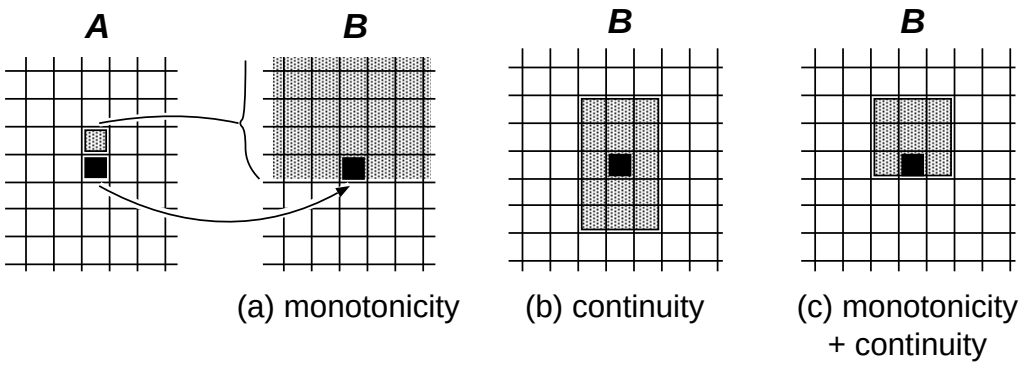


図 2.2: 単調性 ・連続性条件

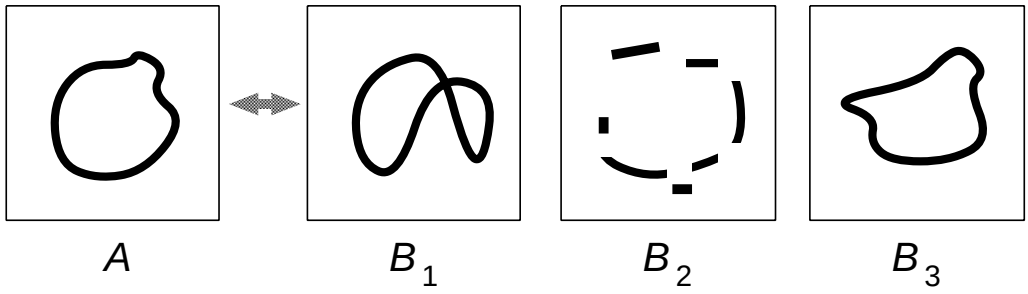


図 2.3: 単調連続性の効果を説明するための例

問題が生じる．これに対し単調性と連続性を併せ考えると，そのような過変形を回避することができ，パターンの位相構造を近似的に保存する 2 次元ワープが構成できる[3]．

単調連続性条件に加えて次の境界条件を用いる．

$$x(1,j) = y(i,1) = 1, \qquad x(N,j) = y(i,N) = M \tag{2.7}$$

以上の条件 (2.3)~(2.7) は等方性を持つ．すなわち，目的関数 (2.1) が最小化される保証があれば，$\boldsymbol{A}$，$\boldsymbol{B}$の縦横を同時に $i \leftrightarrow j$，$x \leftrightarrow y$と交換しても原理的に同じワープが求まる．

[3]厳密に言えば，連続性を図 2.2(b) のように 1 画素分の跳びを許すように定義しているため，$\boldsymbol{B}$ 上の 2 値パターンと$\widetilde{\boldsymbol{B}}$上の 2 値パターンは 4 近傍や 8 近傍の意味で同相にはならない．

第 3 章

動的計画法

3.1 動的計画法の概要

動的計画法 (DP) [5]~[7] は変分問題の数値解法として重要であると同時に，巡回セールスマン問題 (TSP) などの組合わせ最適化問題の代表的な解法でもある [2], [3], [6], [47] ．DP の特長については **1.4**でも簡単に触れたが，変分問題の数値解法として近年注目されている決定論的弛緩法 [14], [15] に対する DP の利点を表 3.1にまとめて示す．

DP アルゴリズムは問題を漸化式により表現することで構成される．ここでは DP の応用としてよく知られた音声認識における時間軸整合のための 1 次元ワープ決定問題 [39], [40] を例として，漸化式の導出法を説明する．2 つの 1 次元系列 $\{a(i) \mid i = 1, \ldots, N\}$ および $\{b(x) \mid x = 1, \ldots, M\}$ が与えられた時，その間の最適 1 次元ワープ関数 $x(i) : \{1, 2, \ldots N\} \rightarrow \{1, 2, \ldots M\}$ は次の制約付き最小化問題の解として与えられる．

$$\min_{x(1),x(2),\ldots,x(N)} \sum_{i=1}^{N} |a(i) - b(x(i))| \tag{3.1}$$

単調連続性条件: $$0 \leq x(i) - x(i-1) \leq 2 \tag{3.2}$$

境界条件: $$x(1) = 1, \quad x(N) = M \tag{3.3}$$

ここで

$$d(i, x(i), x(i-1)) = \begin{cases} |a(i) - b(x(i))| & \text{if } x(i) \text{ and } x(i-1) \text{ satisfy } (3.2), (3.3) \\ \infty & \text{otherwise} \end{cases}$$

を考えると，上の制約付き最適化問題は，次の無制約最適化問題

$$\min_{x(1),x(2),\ldots,x(N)} \sum_{k=1}^{N} d(k, x(k), x(k-1)) \tag{3.4}$$

表 3.1: 決定的弛緩法に対する DP の利点

DP の利点	決定的弛緩法の欠点
解の最適性が保証される．	Eular-Lagrange 方程式は極値の必要条件ではあるが十分条件ではない．また変数の定義域の端点で最大最小となる場合には対応できない．
目的関数の微分可能性の制約が無い．	目的関数が微分可能でないと，Eular-Lagrange 方程式が導き出せない．
様々な制約条件の取り扱いが可能．	制約条件の取り扱いは不可能ではないが，未定定数や，スラック変数の導入が必要になり，結果として計算量が増加する．
演算誤差の集積による不安定性が無い．	本来は整数値変数であっても反復計算中には実数値変数と見なされるため補間誤差が生じる．

に書き直すことができる．ただし $x(0) = 1$ とする．この問題を総当たり法で解こうとすれば，$O(M^N)$ (制約条件を考慮しても$O(3^N)$) の時間計算量が必要となる．

(3.4) 式は以下のように書き直せる．

$$
\begin{aligned}
&\min_{x(1),x(2),\ldots,x(N)} \sum_{k=1}^{N} d(k, x(k), x(k-1)) \\
&= \min_{x(2),\ldots,x(N)} \left[\min_{x(1)} \{ d(1, x(1), x(0)) + d(2, x(2), x(1)) \} + \sum_{k=3}^{N} d(k, x(k), x(k-1)) \right] \\
&= \min_{x(2),\ldots,x(N)} \left[g(2, x(2)) + \sum_{k=3}^{N} d(k, x(k), x(k-1)) \right] \\
&= \cdots = \min_{x(i),\ldots,x(N)} \left[g(i, x(i)) + \sum_{k=i+1}^{N} d(k, x(k), x(k-1)) \right] \\
&= \cdots = \min_{x(N)} g(N, x(N))
\end{aligned}
$$

ここで

$$
g(i, x(i)) = \min_{x(i-1)} \left[g(i-1, x(i-1)) + d(i, x(i), x(i-1)) \right] \tag{3.5}
$$

である．この (3.5) 式は一般に DP 漸化式と呼ばれる．この DP 漸化式を $i = 1$ から順に $i = N$までですべての $x(i)$ について計算するという手続きにより，目的関数の最

小値を得るアルゴリズムが構成される．(3.5) 式の 1 回あたりの計算量が高々$O(M)$であることから，この DP アルゴリズム全体の時間計算量は $O(NM^2)$ で済むことがわかる．

(3.2) 式の $x(i-1)$ と $x(i)$ の相互関係を考慮すれば，DP 漸化式 (3.5) は次のように簡略化できる．

$$\begin{aligned} g(i,x(i)) &= \min_{x(i-1)\in\{x(i),x(i)-1,x(i)-2\}} [g(i-1,x(i-1)) + d(i,x(i),x(i-1))] \\ &= |a(i) - b(x(i))| + \min_{x(i-1)\in\{x(i),x(i)-1,x(i)-2\}} g(i-1,x(i-1)) \end{aligned} \tag{3.6}$$

これにより更に精密な計算量オーダー$O(3NM) = O(NM)$ が得られる．

ここで注意すべきは同じ問題でもその DP 漸化式は一意でない点である．例えば同じ 1 次元ワープ決定問題について次の DP 漸化式も考えられる．

$$\begin{aligned} & g(i,x(i),x(i-1)) \\ &= |a(i) - b(x(i))| + |a(i-1) - b(x(i-1))| \\ &\quad + \min_{\substack{x(i-2)\in\{x(i-1),x(i-1)-1,x(i-1)-2\} \\ x(i-3)\in\{x(i-2),x(i-2)-1,x(i-2)-2\}}} g(i-2,x(i-2),x(i-3)) \end{aligned} \tag{3.7}$$

この DP 漸化式を $i = 2$ から Nまで 1 つおきに計算すれば，(3.6) 式を用いた場合と同一の解が求まる．

3.2 多段決定過程

多段決定過程 (multistage decision process) は DP と密接な関係を持つ概念の 1 つである．Bellman [5] 流の用語に従えば，多段決定過程は段 (stage)，状態 (state)，決定 (decision)，状態遷移 (state transition)，および遷移コスト (transition cost) によって記述され，また一般に有限状態オートマトンのように状態図表現される．

前節のような手続きによりある問題についてそれを解く DP 漸化式が得られれば，その問題を多段決定過程により直ちに表現できる．例えば前出の 1 次元ワープ決定問題の場合，DP 漸化式として (3.6) 式を用いると，$i = 1,\ldots,N$が段に対応する．第 i 段では $a(i)$ のマッピング $x(i)$ が決定される．この $x(i)$ として許されるものの集

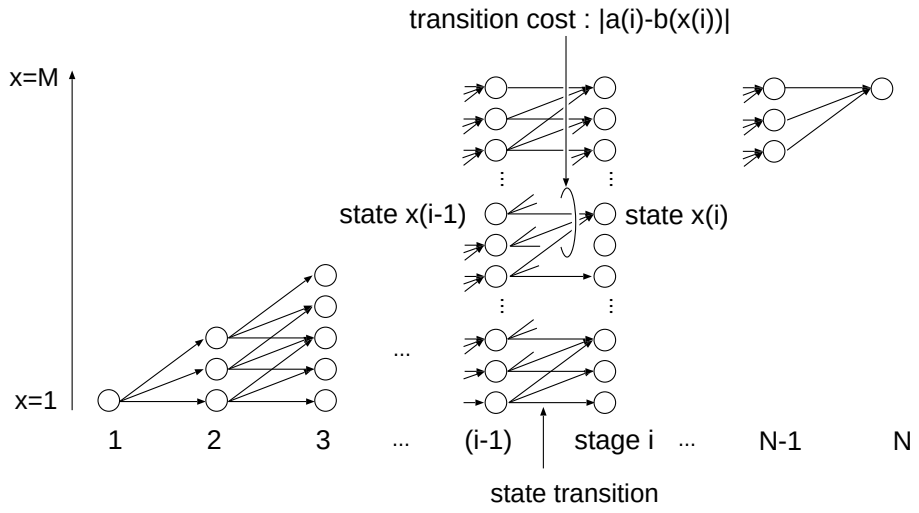


図 3.1: 1 次元ワープ決定問題 [39], [40] を表現する多段決定過程

合 $\{x(i)\}$ が第 i 段に属する状態の集合となる．すなわち，第 i 段での決定 $i \to x(i)$ の結果，状態は $x(i)$ へ遷移する[1]．第 $i-1$ 段の状態 $x(i-1)$ と第 i 段の状態 $x(i)$ が制約条件 (3.2) を満たす場合， $x(i-1)$ から $x(i)$ へ状態遷移できる．最後に状態 $x(i)$ への遷移に遷移コスト $|a(i)-b(x(i))|$ を加法的に与える．こうして定義された 1 次元ワープ決定問題を表現する多段決定過程を図 3.1に示す．

以上の DP 漸化式と多段決定過程の対応関係より明らかなように，多段決定過程の各状態について，そこに遷移可能な前段の状態のうち累積コストの最小のもの選択しそれに遷移コストを加算することで DP 漸化式計算が実現する．この処理を初期状態から最終状態まで行なえばその最少累積コストとして目的関数の最小値が得られ，さらにバックトラック処理により最適状態遷移系列 (決定列) が与えられる．

DP アルゴリズムの計算量は対応する多段決定過程の規模から見積もることができる．DP アルゴリズムの時間計算量はその多段決定過程中の総状態遷移数に比例する．これは段数，1 段当たりの状態数，1 状態から可能な遷移数の積で与えられ

[1] この場合，決定と状態は同一視できる．しかし 5章のピクセルワイズ DP アルゴリズムのように，決定と状態が同一視できない場合もある．

る．一方，空間計算量は総状態数 (バックトラック処理が不要な場合，1 段あたりの状態数) に比例する．これは段数と 1 段当たりの状態数の積で与えられる．

ある問題に対する DP 漸化式が一意でないことから明らかなように，多段決定過程も一意ではない．例えば (3.7) 式の DP 漸化式に対する ($N/2$ 段の) 多段決定過程も構成することができる．

本論文では多段決定過程を用いて DP アルゴリズムの説明を行なう．DP アルゴリズムの議論をする上で多段決定過程を用いることの利点は，状態図表現により問題の構造の見通しがよくなる点と，以上に述べたようにアルゴリズムの時間および空間計算量の見積りが容易になるという点である．

なお，多段決定過程は DP に固有の概念ではなく，問題を木探索や総当たり法で解く場合も構成することができる．例えば木探索の場合の多段決定過程は木構造となり，図 3.1のようないわゆるトレリス構造にはならない．

3.3 ビームサーチ

DP アルゴリズムは計算量が多いことが欠点と言われる．もちろん総当たり法に比べると DP は洗練された方法ではあるが，「次元の呪い (curse of dimensionality)」と言われるように，その計算量は問題のある種の困難さに鋭敏に反応し，結果として指数オーダーの計算量を持つアルゴリズムとなる場合がある．このため DP を用いる場合には，その計算量を常に考慮する必要がある．

解の最適性を保ったまま計算量を減らす方法としては，分枝限定法の限定操作の概念を採り入れ，最適状態系列に含まれる可能性が完全に無い状態を除外する方法 [2], [48] がある．しかし計算量の少なくて済む単純な下界値関数では効率のよい限定操作は期待できず，大きな改善は見込めない．

一方，解の最適性をあきらめ，DP の近似アルゴリズムが検討されることも多い [47], [49]∼[56]．それら近似アルゴリズムのうち一般的なものとしてビームサーチ DP アルゴリズム [53], [54] が挙げられる．ビームサーチ (beam search) とは，局所的な判定基準により最適決定系列に含まれる可能性が低いと判断された状態を以後の

処理から除外するという，いわゆる枝刈処理 (pruning) を各段で行なうことにより探索空間を圧縮し，計算量の低減を実現する効率化手法である．このビームサーチと DP アルゴリズムを組合せはビームサーチ DP アルゴリズムと呼ばれる．ビームサーチの適用により解の最適性の保証はなくなるが， 実用的にはその近似解で十分なことが多い [47], [53]∼[56] ．

ビームサーチ DP アルゴリズムでは，ビームサーチを効率的なものとするために，動的リストとビーム内データ駆動方式が用いられることが多い [53]∼[55] ．動的リストとは，枝刈の結果残った状態だけをメモリ上にリスト化して格納する方法で，これにより空間計算量を大幅に低減することができる．ビーム内データ駆動方式とは，リスト上に残った状態から DP 漸化式を駆動する手法で，これにより時間計算量を削除することができる．

ビームサーチ DP アルゴリズムで用いられる枝刈判定処理としてはしきい値処理が一般的である．ここで用いるしきい値としては，(a) 固定値，(b) その段の累積コストの最小値に余裕定数を加えたもの，(c) その段の累積コストのうち R 番目に小さい値θ，のいずれかが用いられる． 本研究では (c) を用いた．この理由は，(a) や (b) のしきい値では計算量の見積りが難しく，また状態が非常に多くなる問題においては実行可能な程度まで枝刈が行なわれない場合が発生するためである．以下，Rをビーム径と呼ぶ．

近似アルゴリズムは精度保証 [4], [57] ができることが望ましいが，ビームサーチ DP アルゴリズムではビーム径 Rと近似精度の間に単調な関係は保証されない．しかし平均的にはビーム径が大きいほど近似精度は高くなると考えられる．実際，本論文で示す実験結果 (**4.6.2** , **5.6.3** , **8.3**) もその傾向を示している．

第 4 章
動的計画法による解法 1 – カラムワイズ DP アルゴリズム

4.1　まえがき

本章では，第 2章で定義した単調連続 2 次元ワープ決定問題，すなわち (2.2)∼(2.7) の最適化問題の DP による解法を検討する．多段決定過程としての定式化が必要になるが，そこでは段，状態，状態遷移をどのように考えるかが問題となる．これに対し，ワープ全体の決定問題を列単位の部分問題に分解して捉えるという一つの自然な考え方から，列の推移の各段で各列のマッピングを決定するという逐次過程としての定式化が可能となる．この場合，列内の単調連続性制約が各段に属する状態を規定し，また列間の単調連続性制約が可能な状態遷移を規定することになる．本章で提案するカラムワイズ DP アルゴリズムは，こうして定義される多段決定過程の最適状態遷移系列を与えるものである．

なお，付録 **A.1**に，カラムワイズ DP アルゴリズムが正しく単調連続 2 次元ワープ決定問題を解くことの証明を与えた．

4.2　列を処理単位とする多段決定過程による問題表現

4.2.1　段，決定および状態

画像$\boldsymbol{A}$の第 i 列を

$$\boldsymbol{a}(i) = (a(i,1), \ldots, a(i,j), \ldots, a(i,N)) \tag{4.1}$$

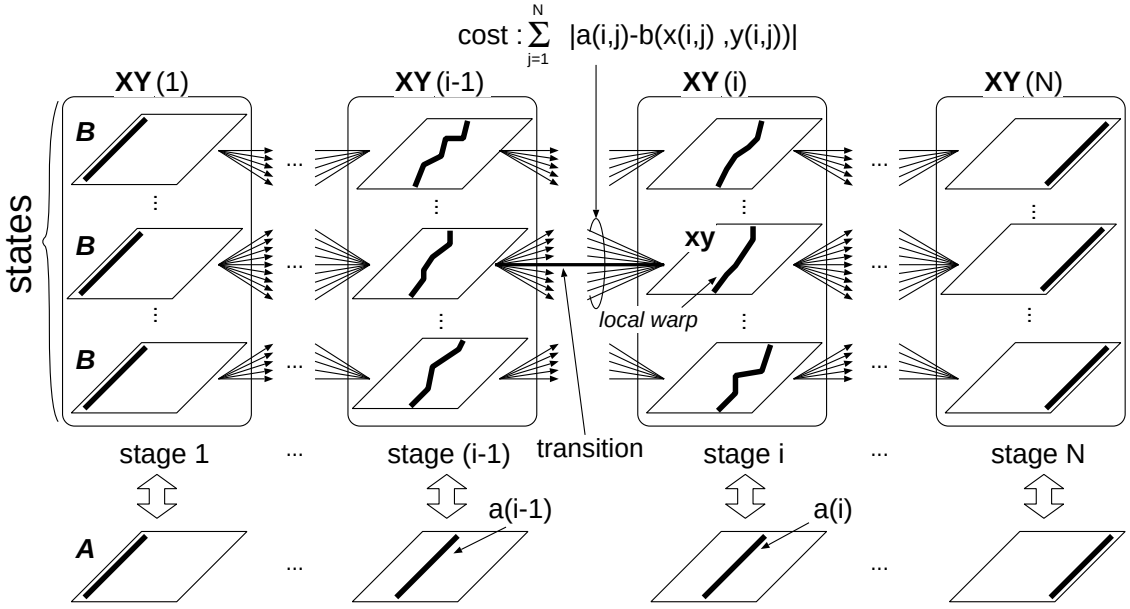


図 4.1: 単調連続 2 次元ワープ決定問題を表現するカラムワイズな多段決定過程

と表す．ここで $i = 1$ から Nまでの N段の決定過程を考え，その第 i 段では$\boldsymbol{a}(i)$ のマッピングが決定されるとする．決定が列 (column) を単位としてなされるので，これをカラムワイズな多段決定過程と呼ぶ．図 4.1にこの多段決定過程の状態図表現を示す．

第 i 段に属する各状態は $\boldsymbol{a}(i)$ の 1 つのマッピングに対応し，このため決定と状態は同一視することができる．このマッピングによる $\boldsymbol{a}(i)$ の $\boldsymbol{B}$ 上での像を局所ひずみパターンと呼び，$\boldsymbol{xy}(i)$ で表す $(\text{図 } 4.2(\mathrm{a}))$．

$$\boldsymbol{xy}(i) = ((x(i,1), y(i,1)), \ldots, (x(i,j), y(i,j)), \ldots, (x(i,N), y(i,N))) \qquad (4.2)$$

以下では記号 $\boldsymbol{xy}(i)$ により，その局所ひずみパターンが対応する状態も表わす．図 4.1の状態図では，各状態をその局所ひずみパターンにより具体的に表現している．

局所ひずみパターン$\boldsymbol{xy}(i)$ としては，単調連続性条件 (2.4)，(2.5)，および境界条件 (2.7) を満たす範囲で様々な形状が選べる．その集合を$\boldsymbol{XY}(i) = \{\boldsymbol{xy}(i)\}$ と表す．なお，残りの単調連続性条件 (2.3)，(2.6) は，次節で述べるように状態遷移に対する制約となる．

4.2.2　状態遷移

上述のように，状態$\boldsymbol{xy}(i)$と$\boldsymbol{xy}(i-1)$はそれぞれ$\boldsymbol{a}(i)$と$\boldsymbol{a}(i-1)$の1つのマッピングに対応する．よって状態$\boldsymbol{xy}(i)$と$\boldsymbol{xy}(i)$へ遷移可能な状態$\boldsymbol{xy}(i-1)$は互いに単調連続性(2.3)，(2.6)を満たす必要がある．例えば，図4.2(a)の$\boldsymbol{xy}(i)$を同図(b)のようにxおよびy方向へ分解したものを$\boldsymbol{x}(i)$，$\boldsymbol{y}(i)$とすれば，同図(c)の灰色の部分に含まれる$\boldsymbol{x}(i-1)$，$\boldsymbol{y}(i-1)$をx，y成分とする$\boldsymbol{xy}(i-1)$が，$\boldsymbol{xy}(i)$へ遷移可能な状態である．以下$\boldsymbol{xy}(i)$に遷移可能な$\boldsymbol{xy}(i-1)$の集合を$\overline{\boldsymbol{XY}}(\boldsymbol{xy}(i))$と表す．

4.2.3　遷移コスト

状態$\boldsymbol{xy}(i)$への遷移は，$\boldsymbol{a}(i)$を局所ひずみパターン$\boldsymbol{xy}(i)$により$\boldsymbol{B}$上へのマッピングすることの決定に相当する．よって，(2.2)の目的関数を考慮すれば状態$\boldsymbol{xy}(i)$への遷移に対するコストは次式で与えられる．

$$d(i,\boldsymbol{xy}(i)) = \sum_{j=1}^{N} |a(i,j) - b(x(i,j), y(i,j))| \tag{4.3}$$

第1段から第N段までのある状態遷移系列(決定列)に対してこの遷移コストを累積すれば，その状態遷移系列に対応する単調連続2次元ワープの目的関数値Dを計算することができる．

4.3　カラムワイズ DP アルゴリズム

最適な単調連続2次元ワープは，以上で定義された多段決定過程上のすべての状態遷移系列のうち最小累積コストを持つものに対応する．図4.3にその最適状態遷移系列を探索するDPアルゴリズム(カラムワイズDPアルゴリズム)を示す．図中の$g(i,\boldsymbol{xy})$は，ある初期状態$\boldsymbol{xy}(1)$から状態$\boldsymbol{xy} \in \boldsymbol{XY}(i)$までの，最適状態遷移系列に対する累積コストを表す．Step 5がDP漸化式に相当する．なお$D(\boldsymbol{A},\boldsymbol{B})$だけでなく最適2次元ワープ関数を具体的に求める必要がある場合，この他にバックポインタやバックトラック処理が必要になるが，それらの実装は容易である．

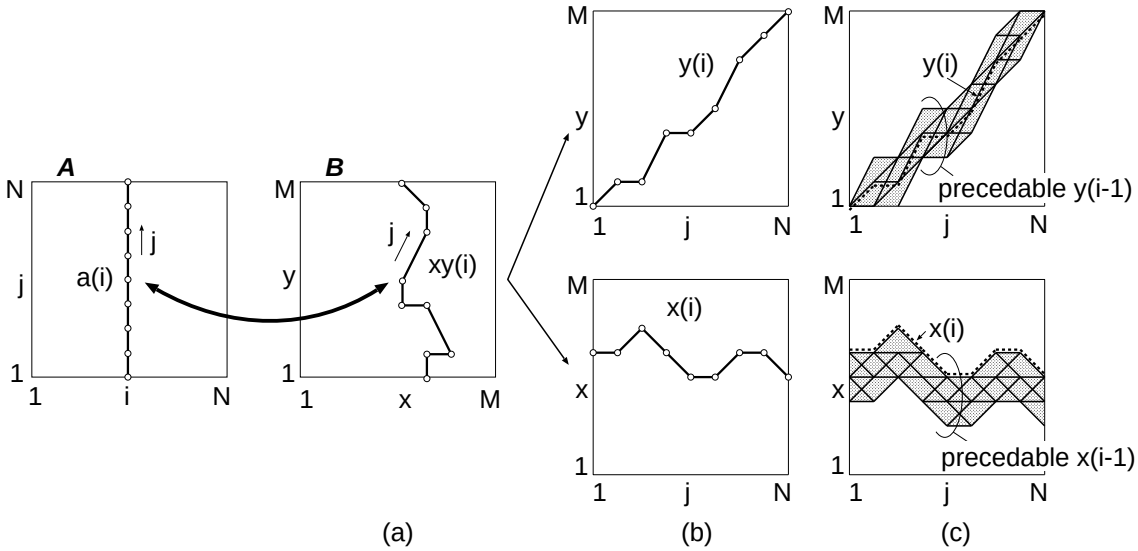


図 4.2: (a) 局所ひずみパターン $\boldsymbol{xy}(i)$ の例，(b) その x，y 方向への分解，(c) $\overline{\boldsymbol{XY}}(\boldsymbol{xy}(i))$ に含まれる $\boldsymbol{xy}(i-1)$ の x，y 方向での領域

/* *Initialization* */
1 **for all** $\boldsymbol{xy} \in \boldsymbol{XY}(1)$ **do**
2 $\quad g(1, \boldsymbol{xy}) = d(1, \boldsymbol{xy})$

/* *Recursion* */
3 **for** $i := 2$ **to** N **do**
4 $\quad$ **for all** $\boldsymbol{xy} \in \boldsymbol{XY}(i)$ **do**
5 $\quad\quad g(i, \boldsymbol{xy}) = d(i, \boldsymbol{xy}) + \min_{\overline{\boldsymbol{xy}} \in \overline{\boldsymbol{XY}}(\boldsymbol{xy})} g(i-1, \overline{\boldsymbol{xy}})$

/* *Termination* */
6 $D(\boldsymbol{A}, \boldsymbol{B}) = \min_{\boldsymbol{xy} \in \boldsymbol{XY}(N)} g(N, \boldsymbol{xy})$

図 4.3: カラムワイズ DP アルゴリズム

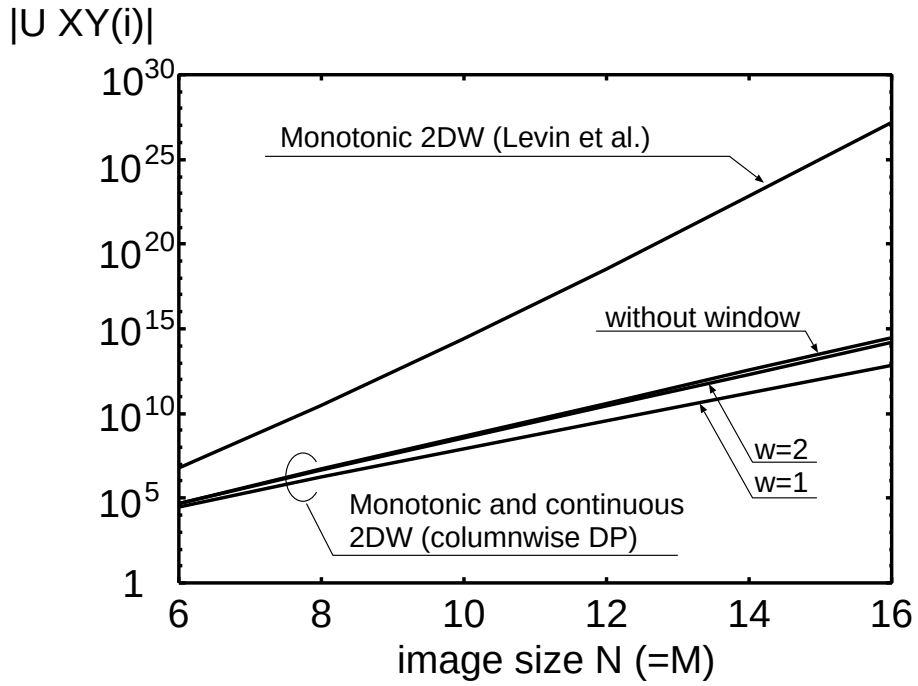


図 4.4: 局所ひずみパターンの総数 $|\bigcup_i \boldsymbol{XY}(i)|$

4.4 計算量の評価

本節では，$N = M$の場合の，カラムワイズ DP アルゴリズムの時間計算量および空間計算量を考察する．**3.2**で述べたように，DP に基づくアルゴリズムの時間計算量は多段決定過程中の総遷移数に比例し，空間計算量は多段決定過程中の全状態数 (バックトラックを必要しない場合は 1 段当たりの状態数) に比例する．

まず 1 段当たりの状態数 $|\boldsymbol{XY}(i)|$ を見積もる．局所ひずみパターンをその絶対位置 $(x(i,1), y(i,1))$ と形状に分離して考える．境界条件 (2.7) により $y(i,1) = 1$ であることを考えれば，取り得る $(x(i,1), y(i,1))$ の数は $N = O(N)$ となる．また局所歪みパターンの形状は単調連続性条件 (2.4)，(2.5) により$O(9^N)$ 通りの可能性がある．よって $|\boldsymbol{XY}(i)| = O(N) \times O(9^N) = O(N9^N)$ が得られる．参考のため，図 4.4 に$\bigcup_i |\boldsymbol{XY}(i)|$ の数値シミュレーションによる実測値を示す．

次に 1 つの状態から可能な遷移数 $|\overline{\boldsymbol{XY}}(\boldsymbol{xy}(i))|$ を見積もる．単調連続性により，$\boldsymbol{xy}(i-1)$ とそれに隣接可能な$\boldsymbol{xy}(i)$ の jにおける差分 $(x(i,j)-x(i-1,j), y(i,j)-y(i-1,j))$ は，$\boldsymbol{xy}(i-1)$，$\boldsymbol{xy}(i)$ の形状とは無関係に $(\{2,1,0\},\{1,0,-1\})$ で表される 9 種類のいずれかとなる．逆にこの差分 N個の列と$\boldsymbol{xy}(i-1)$ を用意すれば，隣接可能な$\boldsymbol{xy}(i)$ が 1 つ定まる (図 4.2(c))．こうした差分の列の全部は，各ノードがそれぞれ 9 種類の差分の 1 つに対応する子ノードを持つような深さ Nの 9 分木によって表現できる．すなわちこの 9 分木の根から葉までの 9^N本の道により，$\boldsymbol{xy}(i-1)$ に隣接可能な$\boldsymbol{xy}(i)$ の全部が表現される．よって $|\overline{\boldsymbol{XY}}(\boldsymbol{xy}(i))| = O(9^N)$ が得られる．

以上の見積りと段数が Nであることより，カラムワイズ DP アルゴリズムの時間計算量として $N \times O(N9^N) \times O(9^N) = O(N^2 9^{2N})$ が得られる．また空間計算量として，バックトラックにより最適ワープ関数を求める必要がある場合について $N \times O(N9^N) = O(N^2 9^N)$，バックトラックを必要しない場合について $O(N9^N)$ が得られる．

4.5 近似アルゴリズム – カラムワイズビームサーチ DP アルゴリズム

前節で述べたように，カラムワイズ DP アルゴリズムの時間空間量および空間計算量は，共に Nに関して指数オーダーで増加する．このためカラムワイズ DP アルゴリズムをそのまま現実的なサイズの画像に適用することは事実上不可能である．そこで，ビームサーチの適用による近似アルゴリズムを検討する．

3.3で述べたように，ビームサーチは基本的に各段で最適状態系列に含まれる可能性の低い状態を枝刈により除外する処理により計算量を低減する方法である．しかしこの定義に従ってビームサーチをカラムワイズ DP アルゴリズムにそのまま適用しても多項式時間計算量を持つビームサーチ DP アルゴリズムは得られない[1]．これは各状態が遷移可能な状態数 $|\overline{\boldsymbol{XY}}(\boldsymbol{xy}(i))|$ が指数オーダーであることに因る．

[1]明らかに，多項式時間アルゴリズムの空間計算量は多項式オーダーである．ただし逆は成り立たない．

```
    /* Sub-transition (j = 1) */
 1  l := 1
 2  for t := 1 to |LIST_{i-1}| do begin
 3    ⟨ḡ, x̄ȳ⟩ := LIST_{i-1}[t]
 4    (bx, null) :=first element of x̄ȳ
 5    for x := bx to bx + 2 do begin
 6      NODE_1[l] := ⟨(ḡ + |a(i,1) − b(x,1)|), x, 1, null, t⟩
 7      l := l + 1
 8    end
 9  end

    /* Sub-transitions (j = 2, ..., N) and pruning */
10  for j := 2 to N do begin
11    l := 1
12    θ := Rth smallest cost g in NODE_{j-1}    /* Deciding pruning threshold */

    /* Scanning NODE_{j-1} */
13    for k := 1 to |NODE_{j-1}| do begin
14      ⟨ḡ, ax, ay, null, t⟩ := NODE_{j-1}[k]
15      if ḡ > θ then continue    /* Pruning */

        /* Creating new nodes and storing to NODE_j */
16      ⟨null, x̄ȳ⟩ := LIST_{i-1}[t]
17      (bx, by) := jth element of x̄ȳ
18      for all (x, y) which satisfy monotonicity, continuity and boundary
                conditions with respect to (ax, ay) and (bx, by) do begin
19        NODE_j[l] := ⟨(ḡ + |a(i,j) − b(x,y)|), x, y, k, t⟩
20        l := l + 1
21      end
22    end
23  end
```

図 4.5: カラムワイズビームサーチ DP アルゴリズムの第 i 段での処理

```
    /* Merging paths of the same xy into one with their smallest g */
24  θ := Rth smallest cost g in NODE_N
25  t := 1
26  for k := 1 to |NODE_N| do begin
27    ⟨g, x, y, l, null⟩ := NODE_N[k]
28    if g > θ then continue    /* Pruning */
29    Obtaining xy through backtracking from NODE_{N-1}[l] to NODE_1
30    Searching LIST_i for the item with xy
31    if the item with xy is found at LIST_i[p] then begin    /* Selection */
32        ⟨g', null⟩ := LIST_i[p]
33        if g < g' then LIST_i[p] := ⟨g, xy⟩
34    end
35    else begin /* Appending */
36        LIST_i[t] := ⟨g, xy⟩
37        t := t + 1
38    end
39  end
```

図 4.5: (続き) カラムワイズビームサーチ DP アルゴリズムの第 i 段での処理

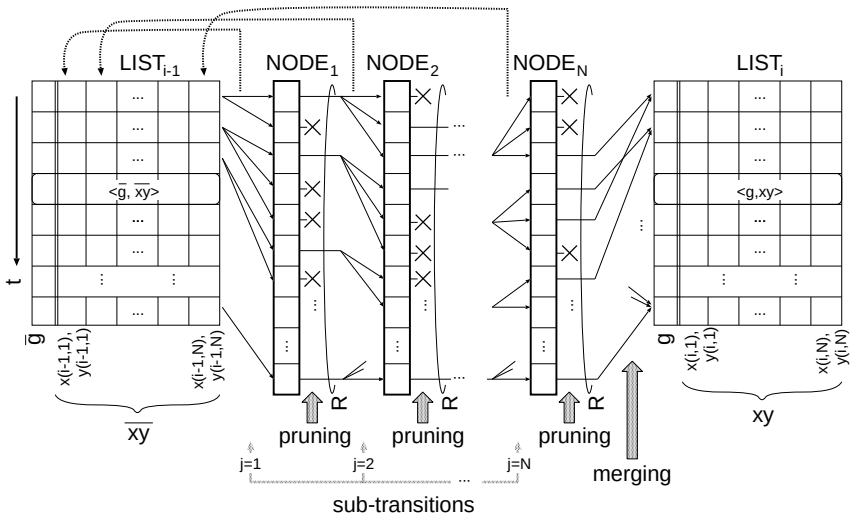


図 4.6: 部分遷移毎の枝刈処理を伴う DP 漸化式処理

そこで，ビーム内に残った状態$\boldsymbol{xy}(i-1)$より**4.4**で述べた$\boldsymbol{xy}(i-1)$が遷移可能なすべての$\boldsymbol{xy}(i)$を表現する9分木をそれぞれ駆動し，すべての木について並列的に枝を伸ばしていくことを考える．このとき木の深さjを1つ増す毎に，累積コスト

$$g(i-1,\boldsymbol{xy}(i-1))+\sum_{k=1}^{j}|a(i,k)-b(x(i,k),y(i,k))|$$

が最小のR個のノードだけを残す枝刈処理を行なう．これは遷移$\boldsymbol{xy}(i-1)\rightarrow\boldsymbol{xy}(i)$を$N$回の部分遷移(sub-transition)に分割したことに相当する．この枝刈処理を$j=1$からNまでくり返した後，同じ$\boldsymbol{xy}(i)$を表す葉が複数存在すればその中で最小の累積コストを持つものを選択することで第i段での処理が終了する．

このアルゴリズムをカラムワイズビームサーチDPアルゴリズムと呼ぶ．図4.5にその第i段での処理を示す．ここでは第$i-1$段での処理の結果残っている状態$\overline{\boldsymbol{xy}}\in\boldsymbol{XY}(i-1)$がその状態までの累積コスト$\overline{g}$と共に動的リスト$LIST_{i-1}$に格納されていると仮定している．このリストの構成要素$\langle\overline{g},\ \overline{\boldsymbol{xy}}\rangle$をアイテムと呼ぶ．$LIST_{i-1}$の各アイテムから9分木を駆動しているので，カラムワイズビームサーチDPアルゴリズムも変則的ながらビーム内データ駆動方式(**3.3**参照)を用いている．ただし前述のようにDP漸化式はN回の部分遷移に分けて計算される(図4.5 Step 1-23)．部分遷移の結果はリスト$NODE_j$に格納される．このリスト$NODE_j$中の各アイテムは5つの項目からなり，$LIST_i$の生成に必要な情報などを持つ．部分遷移毎に$NODE_{j-1}$に対し枝刈を行なうことで(Step 15)，$NODE_j$のアイテム数を高々$9R$個に押えており，これにより多項式時間計算量のアルゴリズムが実現している．部分遷移が$j=N$まで終了した後，$NODE_N$中の同じ$\boldsymbol{xy}(i)$を表すアイテムを1つに統合しながら$LIST_i$を生成することで(Step 24-39)，第i段での処理が終了する．図4.6にリスト$LIST$，$NODE$，およびその上での処理の様子を示す．

4.6 計算量の評価

4.6.1 時間および空間計算量

第 i 段では，画素間距離加算$O(R)$，枝刈基準決定[2]$O(R)$ を要する部分遷移が N 回繰り返された後 (図 4.5 Step1-23)，最小値選択$O(NR^2)$(同図 Step24-39) がなされる．よって，第 i 段の時間計算量は $N \times \{O(R) + O(R)\} + O(NR^2) = O(NR^2)$ となる．アルゴリズム全体ではこの処理が N段分繰り返されるので，結局$O(N^2R^2)$ の時間計算量で単調連続 2 次元ワープ決定問題の近似解が求まる．空間計算量も，バックトラックの必要な場合$O(N^2R)$ に，不要な場合$O(NR)$ に改善される．

4.6.2 計算量と近似精度の関係

カラムワイズビームサーチ DP アルゴリズムの計算量と近似精度の関係を見るために，$[0, 1)$ の一様乱数で作成した画像 100 対を対象として，ビーム径 Rを変えながら $D(\boldsymbol{A}, \boldsymbol{B})$ に関する平均近似誤差 ，および平均計算時間 (Sun Ultra 1 ; SPECint_92: 215, SPECfp_92: 303) を実測した．最適解を計算するために，画像サイズは $N = M = 6$ とした．輝度値をそのまま画素特徴量とし，100 対それぞれについてカラムワイズ DP アルゴリズムとカラムワイズビームサーチ DP アルゴリズムにより最適解と近似解を求め，それから平均誤差を求めた[3]．結果を図 4.7に示す．ビーム径 R を減少させれば計算時間を単調に短縮でき，逆に Rを増加させれば近似精度を単調に向上できることがわかる．このように近似精度と計算時間はトレードオフの関係にあるので，タスクや実行環境に応じて Rを適宜設定すればよい．

[2] R番目に小さな値を求めるためには，$O(R)$ で済む選択法 [3], [58] で十分であって，$O(R \log R)$ を要するソートは必要ない．

[3] 近似解は必ず最適解以上になるので，誤差は (近似解−最適解) で与えられる．

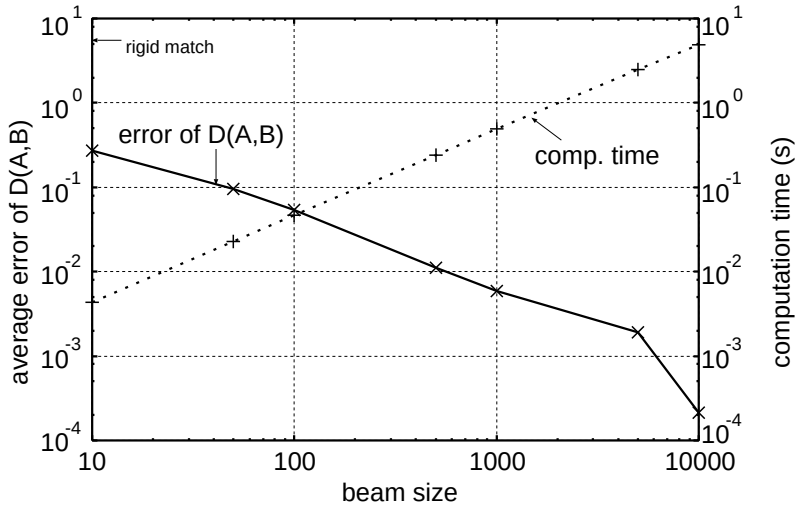


図 4.7: カラムワイズビームサーチ DP アルゴリズムの計算時間および近似誤差の平均値

4.7 まとめ

本章では，最適な単調連続 2 次元ワープ決定問題の DP による解法を検討した．各段で画像の各列のマッピングを決定するという多段決定過程の定式化から，カラムワイズ DP アルゴリズムを導出した．また多段決定過程の規模を見積もることにより，アルゴリズムの時間および空間計算量がそれぞれ$O(N^2 9^{2N})$，$O(N^2 9^N)$ であることを明らかにした．

次に実行可能な範囲までの計算量の低減を目指し，ビームサーチを適用した近似アルゴリズムを検討した．ビームサーチの単純な適用では多項式時間近似アルゴリズムが得られないために，1 回の状態遷移を N回の部分遷移に分解し，各部分遷移毎に枝刈を行なう手法を提案した．その時間および空間計算量が Nに関して多項式オーダーとなることを示した．このアルゴリズムについて，計算量と解の精度の間にトレードオフの関係があることを実験的に示した．

第 5 章
動的計画法による解法 2 – ピクセルワイズ DP アルゴリズム

5.1 まえがき

前章と同様に，第 2章で定義した単調連続 2 次元ワープ決定問題，すなわち (2.2)∼(2.7) の最適化問題の DP による解法を検討する．前章で提案したカラムワイズ DP アルゴリズムは，各段で画像の各列を単位としてマッピングを決定するという考え方を基に設計された多段決定過程に基づくものであった．一方，画像のスキャンに同期して，1 画素毎 (pixelwise) にマッピングを決定していく過程を用いるのも一つの自然な考え方である．この場合，段はスキャン順に従って画素数分 (N^2) 用意される．本章で提案するピクセルワイズ DP アルゴリズムは，こうして定義される多段決定過程の最適状態遷移系列を与えるものである．

ピクセルワイズ DP アルゴリズムは，カラムワイズ DP アルゴリズムを計算量の点で改良したものになっている．両アルゴリズムの計算量の比較は次章で行なう．

なお，付録 **A.2**に，ピクセルワイズ DP アルゴリズムが正しく単調連続 2 次元ワープ決定問題を解くことの証明を与えた．

5.2 画素を処理単位とする多段決定過程による問題表現

5.2.1 段および決定

画像$\boldsymbol{A}$をラスタスキャンしながら，その順に従って各画素 (i, j) のマッピングを決定していく過程を考える．段は (i, j) に関するラスタスキャン順に用意され，第 (i, j) 段では画素 (i, j) のマッピング，すなわち $(x(i, j), y(i, j))$ が決定される．決定

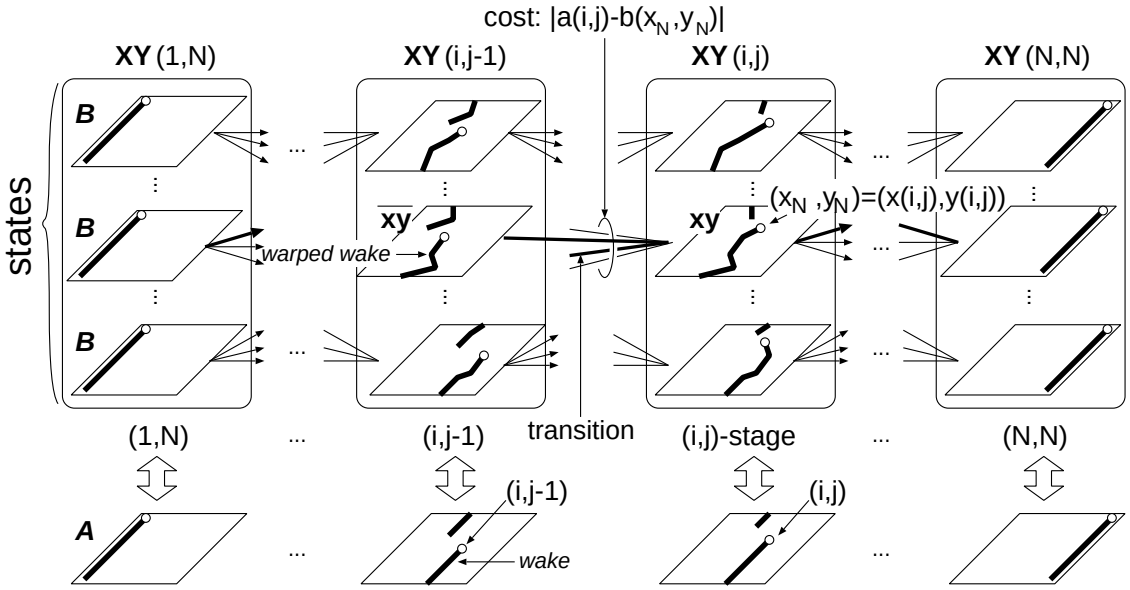


図 5.1: 単調連続 2 次元ワープ決定問題を表現するピクセルワイズな多段決定過程

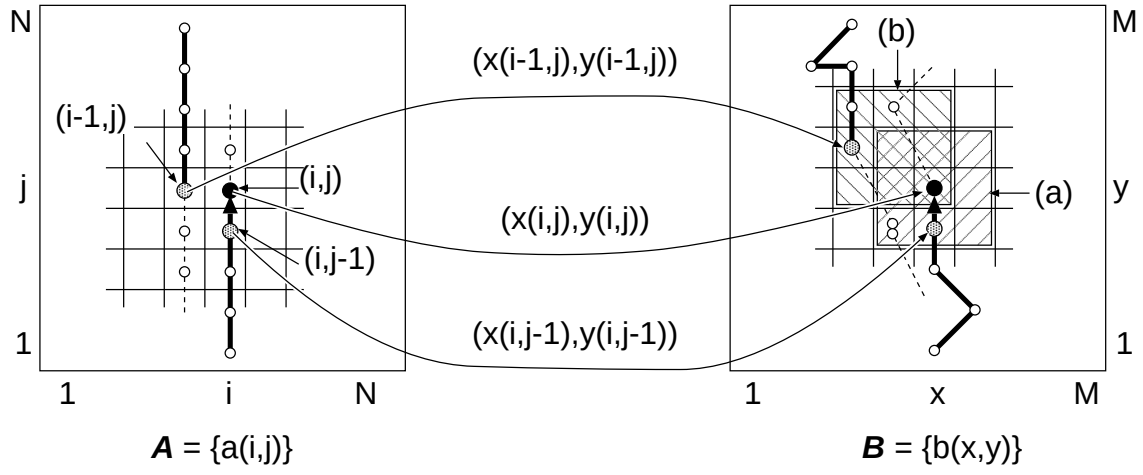


図 5.2: ラスタスキャン順の決定過程とその N 次マルコフ性

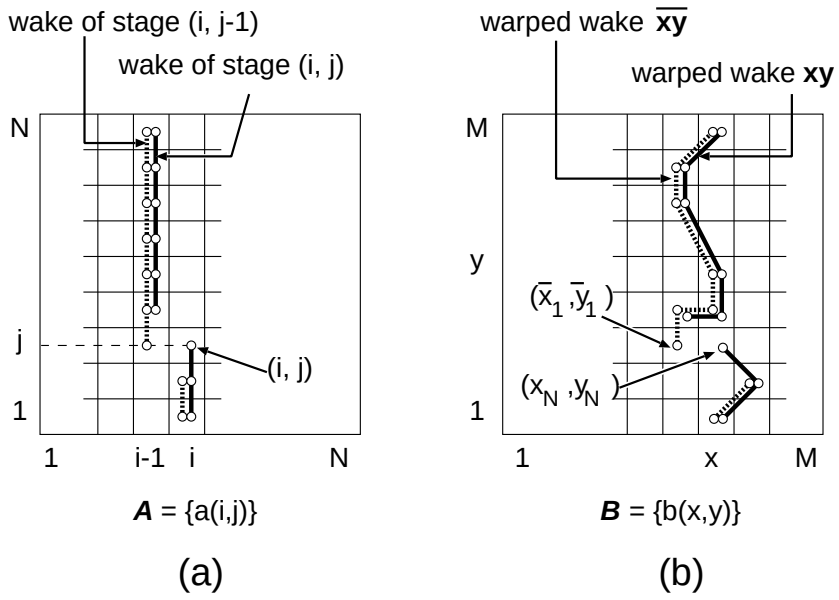


図 5.3: ウェイクとウェイク像

が画素を単位としてなされるので，これをピクセルワイズな多段決定過程と呼ぶ．図 5.1にピクセルワイズ DP アルゴリズムに対応する多段決定過程を示す．

各段での決定は (2.3)〜(2.6) の単調連続性条件により互いに制約される $(図\ 5.2)$．条件 (2.4)，(2.5) により，$(x(i,j), y(i,j))$ は直前に決定された $(x(i,j-1), y(i,j-1))$ に依存し，図 5.2中に斜線で示された大きさ 3×3 の領域 (a) に制限される．さらに条件 (2.3)，(2.6) により，$(x(i,j), y(i,j))$ は N段前に決定された $(x(i-1,j), y(i-1,j))$ に依存し，同図の 3×3 領域 (b) に制限される．図 5.2の場合，2 つの領域 $(\mathrm{a})(\mathrm{b})$ が交差している 4 画素が $(x(i,j), y(i,j))$ として決定可能な領域となる．このように第 (i,j) 段での決定は直前の $(i,j-1)$ 段での決定に加えて N段前の $(i-1,j)$ 段での決定にも制約を受ける．この N次マルコフ性を扱うためにウェイクの概念を導入する．各 (i,j) 段のウェイク $(\mathrm{wake}:$ 航跡$)$ とは，(i,j) を先頭とするラスタスキャン順の N画素の列 $[(i-1,j+1), \ldots, (i-1,N), (i,1), \ldots, (i,j)]$ である．図 $5.3(\mathrm{a})$ に，第 $(i,j-1)$ 段と，(i,j) 段のウェイクを示す．

なお，例外的に最初の第 $(1,N)$ 段では N画素 $(1,1), \ldots, (1,N)$ のマッピングが

決定されるとする．このマッピングは単調連続性条件 (2.4)，(2.5) および境界条件 (2.7) によって制限される．

5.2.2 状態

第 (i,j) 段の 1 つの状態は，第 (i,j) 段のウェイクの 1 つのマッピングに対応する．以後このマッピングをウェイク像 (warped wake) と呼び，$\boldsymbol{xy}(i,j)$ もしくは単に$\boldsymbol{xy}$と表記する．

$$\begin{aligned}\boldsymbol{xy}(i,j) \;=\; & [(x(i-1,j+1),y(i-1,j+1)),\ \ldots\ ,(x(i-1,N),y(i-1,N)),\\ & (x(i,1),y(i,1)),\ldots\ ,\ (x(i,j),y(i,j))]\end{aligned} \tag{5.1}$$

以下では記号 $\boldsymbol{xy}(i,j)$，$\boldsymbol{xy}$によりそのウェイク像が対応する状態も表わす．状態$\boldsymbol{xy}$へ遷移することは $(i,j)\rightarrow(x(i,j),y(i,j))$ の決定に相当する．

ウェイク像とカラムワイズ DP アルゴリズムの局所歪みパターンは表記上類似しているが本質的に異なるものである．すなわち，局所歪みパターンはその段で決定されるマッピングを表現し，ウェイク像はその段の決定を含め過去の N 段分の決定履歴を表現している．このように図 5.1の多段決定過程では，決定と状態は一致しない．

図 5.3(b) にウェイク像の例を示す．各ウェイク像は，単調連続性条件 (2.4)，(2.5) および境界条件 (2.7) を満たす必要がある．その条件を満たす第 (i,j) 段のウェイク像の集合，すなわち第 (i,j) 段に属する状態の集合 $\{\boldsymbol{xy}(i,j)\}$ を$\boldsymbol{XY}(i,j)$ と表記する．なお，残りの単調連続条件 (2.3)，(2.6) は，次節で議論されるように$\boldsymbol{xy}(i,j-1)$から$\boldsymbol{xy}(i,j)$ への状態遷移に関する条件となる．

5.2.3 状態遷移

第 $(1,N)$ 段から (N,N) 段までの 1 つの状態遷移系列について，各状態の (x_N,y_N) の系列から画像 $\boldsymbol{A}$，$\boldsymbol{B}$ 間の 1 つの単調連続 2 次元ワープが定まる．

状態$\overline{\boldsymbol{xy}} \in \boldsymbol{XY}(i, j-1)$から$\boldsymbol{xy} \in \boldsymbol{XY}(i, j)$に遷移できるためには次の条件を満たす必要がある．

$$\text{単調連続性:} \qquad 0 \leq x_N - \overline{x}_1 \leq 2, \quad |y_N - \overline{y}_1| \leq 1 \tag{5.2}$$

$$\text{整合性:} \qquad x_{k-1} = \overline{x}_k, \quad y_{k-1} = \overline{y}_k \qquad 2 \leq k \leq N \tag{5.3}$$

ここで簡単のため，ウェイク像を以下のように表記した[1]．

$$\boldsymbol{xy} = [(x_1, y_1), \ldots, (x_k, y_k), \ldots, (x_N, y_N)] \tag{5.4}$$

$$\overline{\boldsymbol{xy}} = [(\overline{x}_1, \overline{y}_1), \ldots, (\overline{x}_k, \overline{y}_k), \ldots, (\overline{x}_N, \overline{y}_N)] \tag{5.5}$$

単調連続性 (5.2) は，条件 (2.3)，(2.6) に対応し，図 5.2の領域 (b) の中に (x_N, y_N) が含まれるための条件である．整合性 (5.3) は，ウェイク像 $\overline{\boldsymbol{xy}}$，$\boldsymbol{xy}$ の持つ決定履歴が互いに矛盾しないために必要な条件である．ある $\boldsymbol{xy}$ について以上の条件を満たす $\overline{\boldsymbol{xy}}$ の集合を $\overline{\boldsymbol{XY}}(\boldsymbol{xy}(i, j))$ と表記する．図 $5.3(\mathrm{b})$ の 2 つのウェイク像は，上の条件を満たす 1 つの例になっている．

なお，集合$\overline{\boldsymbol{XY}}(\boldsymbol{xy}(i, j))$ の要素数は最大 9 であり，画像の大きさ (N, M) には依存しない．これは図 5.2中のハッチングされた 2 つの 3×3 領域の重なりが高々9画素であることに因る．

5.2.4 遷移コスト

5.2.1で述べたように，第 (i, j) 段ではマッピング $(x(i, j), y(i, j))$ が決定される．よって，目的関数 (2.2) を考慮して，状態$\boldsymbol{xy} \in \boldsymbol{XY}(i, j)$ への遷移に対して次のコストを与える．

$$\begin{aligned} d(i, j, \boldsymbol{xy}) &= |a(i, j) - b(x_N, y_N)| \quad \text{or} \\ &= |a(i, j) - b(x(i, j), y(i, j))| \end{aligned} \tag{5.6}$$

第 $(1, N)$ 段から第 (N, N) 段までの状態遷移系列に対してこの遷移コストを累積することで，それが対応するの単調連続 2 次元ワープの目的関数値 Dを計算できる．

[1]以下，ウェイク像の表記法として (5.1) と (5.4) のどちらかを適宜選択して用いる．

/* *Initialization* */
1 **for all** $\boldsymbol{xy} \in \boldsymbol{XY}(1,N)$ **do**
2 $\quad g(1,N,\boldsymbol{xy}) := \sum_{k=1}^{N} |a(1,k) - b(x_k,y_k)|$

/* *Recursion* */
3 **for** $i := 2$ **to** N **do begin**

/* $j = 1$ */
4 $\quad$ **for all** $\boldsymbol{xy} \in \boldsymbol{XY}(i,1)$ **do**
5 $\quad\quad g(i,1,\boldsymbol{xy}) := d(i,1,\boldsymbol{xy}) + \min_{\overline{\boldsymbol{xy}} \in \overline{\boldsymbol{XY}}(\boldsymbol{xy})} g(i-1,N,\overline{\boldsymbol{xy}})$

/* $j > 1$ */
6 $\quad$ **for** $j := 2$ **to** N **do**
7 $\quad\quad$ **for all** $\boldsymbol{xy} \in \boldsymbol{XY}(i,j)$ **do**
8 $\quad\quad\quad g(i,j,\boldsymbol{xy}) := d(i,j,\boldsymbol{xy}) + \min_{\overline{\boldsymbol{xy}} \in \overline{\boldsymbol{XY}}(\boldsymbol{xy})} g(i,j-1,\overline{\boldsymbol{xy}})$
9 **end**

/* *Termination* */
10 $D(\boldsymbol{A},\boldsymbol{B}) := \min_{\boldsymbol{xy} \in \boldsymbol{XY}(N,N)} g(N,N,\boldsymbol{xy})$

図 5.4: ピクセルワイズ DP アルゴリズム

5.3 ピクセルワイズ DP アルゴリズム

以上により定義される図 5.1の多段決定過程の第 $(1,N)$ 段から第 (N,N) 段までの状態遷移系列のうち，最小累積コストを与えるものが最適 2 次元ワープに対応する．それを求める DP アルゴリズム (ピクセルワイズ DP アルゴリズム) を図 5.4に示す．$g(i,j,\boldsymbol{xy})$ は，ある初期状態$\boldsymbol{xy}(1,N)$ から状態$\boldsymbol{xy} \in \boldsymbol{XY}(i,j)$ までの最適状態遷移系列に対する累積コストを表す．Step 5 および Step 8 が DP 漸化式に相当する．

なお $D(\boldsymbol{A},\boldsymbol{B})$ だけでなく最適 2 次元ワープ関数を具体的に求める必要がある場

合にはバックポインタやバックトラック処理が必要になるが，それらの実装は容易である．

5.4 計算量の評価

本節では$N = M$の場合のピクセルワイズDPの計算量を考察する．

ピクセルワイズDPアルゴリズムの段数は，**5.2.1**で考察したように$N(N-1) = O(N^2)$である．また1状態から可能な遷移数$|\overline{\boldsymbol{XY}}(\boldsymbol{xy}(i,j))|$は，**5.2.3**で考察したように高々9であり，$O(1)$となる．

1段当たりの状態数は図5.5のようにウェイク像を分割して考えることで見積もることができる．Q_0は可能な$(x(i,1), y(i,1))$の総数である．境界条件(2.7)により$y(i,1) = 1$であることを考えれば，明らかに$Q_0 = N = O(N)$となる．Q_1，Q_2，Q_3は，それぞれ$(x(i,1), y(i,1))$と$(x(i,j), y(i,j))$との間，$(x(i,j), y(i,j))$と$(x(i-1,j+1), y(i-1,j+1))$との間，$(x(i-1,j+1), y(i-1,j+1))$と$(x(i-1,N), y(i-1,N))$との間に部分的なウェイク像を考えた時，それらがとり得る形状の総数である．ウェイク像に対する単調連続性条件(2.4)，(2.5)により，Q_1とQ_3はそれぞれ$O(9^{i-1})$と$O(9^{N-i-1})$である．一方，Q_2は高々25，すなわち定数オーダー$O(1)$となる．これは，図5.2のハッチングされた2つの3×3の領域が重なりを持つような$(x(i-1,j), y(i-1,j))$と$(x(i,j-1), y(i,j-1))$の相対位置の総数である．1段当たりの状態数$|\boldsymbol{XY}(i,j)|$は，以上のQ_0，Q_1，Q_2，Q_3の積で与えられるので，$|\boldsymbol{XY}(i,j)| = O(N9^{i-1}) \times O(1) \times O(9^{N-i-1}) = O(N9^N)$が得られる．なお，この結果はカラムワイズDPアルゴリズムにおける1段当たりの状態数$|\boldsymbol{XY}(i)|$のオーダーと等しい．

以上の見積りと**3.2**の考察より，ピクセルワイズDPアルゴリズムの時間計算量として，$O(N^2) \times O(1) \times O(N9^N) = O(N^3 9^N)$が得られる．また空間計算量として，バックトラックにより最適ワープ関数を求める必要がある場合について$O(N^3 9^N)$，バックトラックを必要しない場合について$O(N9^N)$が得られる．

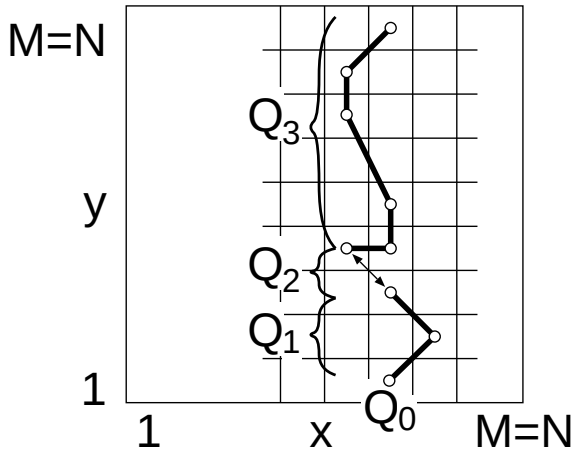


図 5.5: ウェイク像の分割

5.5 近似アルゴリズム – ピクセルワイズビームサーチ DP アルゴリズム

本節では，ピクセルワイズ DP アルゴリズムにビームサーチを適用した，多項式オーダー計算量の近似アルゴリズム (以下，ピクセルワイズビームサーチ DP アルゴリズム) を説明する．ピクセルワイズビームサーチ DP アルゴリズムでは，**3.3** のビームサーチの定義に従って，各段で累積コストの小さなものから R個の状態だけを候補として残す枝刈処理が行なわれており，その結果多項式オーダーの計算量が実現する．これは部分遷移などの工夫が必要であったカラムワイズビームサーチ DP アルゴリズムとは異なる点である．ピクセルワイズビームサーチ DP アルゴリズムでも，ビームサーチを最も効率的なものとするために，動的リストおよびビーム内データ駆動方式が用いられる．それらに加えて，以下に示すように DP 漸化式の最小値選択処理を効率的に行うための工夫がなされている．

図 5.6に第 (i, j) 段での処理を示す．リスト $LIST_{i,j-1}$はいくつかのアイテムより構成される．1 つのアイテムは 1 つの状態に対応し，$\langle \overline{g}, \overline{\boldsymbol{xy}} \rangle$ と記述される．ここで $\overline{\boldsymbol{xy}} \in \boldsymbol{XY}(i, j-1)$ であり，$\overline{g}$ は状態$\overline{\boldsymbol{xy}}$までの累積コストを表す．なお簡単のため

```
    /* Deciding pruning threshold */
 1  θ := Rth smallest cost ḡ in LIST_{i,j-1}

    /* Reset S_{x,y} */
 2  for all x, y do u_{x,y} := 0

    /* Scanning LIST_{i,j-1} */
 3  for t := 1 to |LIST_{i,j-1}| do begin
 4    ⟨ḡ, x̄ȳ⟩ := LIST_{i,j-1}[t]

      /* Pruning */
 5    if ḡ > θ then continue

      /* Creating items */
 6    for all xy which satisfies x̄ȳ ∈ X̄Ȳ(xy) do begin
 7      g := ḡ + d(i, j, xy)

        /* Appending the created item to S_{i,j} */
 8      if u_{x_N,y_N} ≠ 0 then
 9        ⟨g', xy'⟩ := S_{x_N,y_N}[u_{x_N,y_N}]
10      else
11        xy' := null
12      if xy = xy' then /* Selection */
13        if g < g' then
14          S_{x_N,y_N}[u_{x_N,y_N}] := ⟨g, xy⟩
15      else begin /* Append */
16        u_{x_N,y_N} := u_{x_N,y_N} + 1
17        S_{x_N,y_N}[u_{x_N,y_N}] := ⟨g, xy⟩
18      end
19    end
20  end

    /* Make LIST_{i,j} by concatenating S_{i,j} */
21  LIST_{i,j} := {S_{1,1}, ..., S_{x,y-1}, S_{x,y}, ..., S_{M,M}}
```

図 5.6: ピクセルワイズビームサーチ DP アルゴリズムの第 (i, j) 段での処理

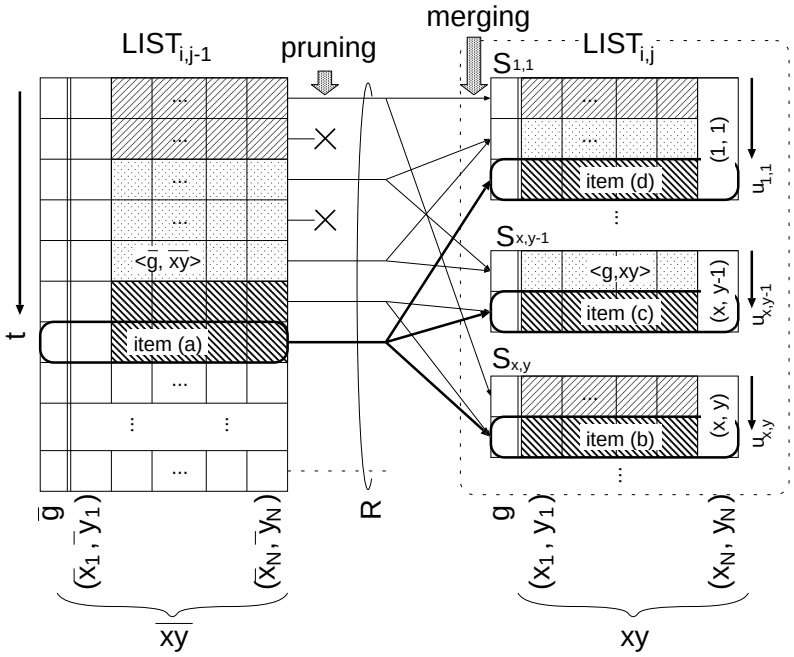


図 5.7: 動的リスト上での DP 漸化式処理

ここではバックポインタを省略している．リスト $S_{x,y}$ は $LIST_{i,j}$ のサブリストであり，$LIST_{i,j-1}$ と同様のアイテム $\langle g, \boldsymbol{xy}\rangle$ (ただし $\boldsymbol{xy} \in \boldsymbol{XY}(i,j)$) により構成される．$S_{x,y}$ に属する $\boldsymbol{xy}$ は $(x_N, y_N) = (x, y)$ を満たすものに限られる．

$\overline{\boldsymbol{xy}}$ の各要素 $(\overline{x}_k, \overline{y}_k)$ を M 進法の数とみなせば，$\overline{\boldsymbol{xy}}$ を $2N$ 桁の M 進数と考えることができる．後に述べるように，一度 $LIST_{1,N}$ の全アイテムをこの数表現に従ってソートしておけば，以後 $LIST_{i,j}$ のアイテムは自動的にソートされるという性質がある．

これらのリスト上で DP 漸化式は次の 2 つの処理により実現される．

1) $LIST_{i,j-1}$ のアイテム $\langle \overline{g}, \overline{\boldsymbol{xy}}\rangle$ から，$\overline{g}$ がしきい値 θ より小さいならば，$\langle g, \boldsymbol{xy}\rangle$ を生成する．ここで，$\boldsymbol{xy}$ は $\overline{\boldsymbol{xy}}$ から遷移可能な状態である ($\overline{\boldsymbol{xy}} \in \overline{\boldsymbol{XY}}(\boldsymbol{xy})$)．(図 5.6 Step 3-8)

2) リスト S_{x_N,y_N} の最後尾のアイテムを $\langle g', \boldsymbol{xy}'\rangle$ とする．もし $\boldsymbol{xy} \neq \boldsymbol{xy}'$ なら

ば，生成されたアイテム $\langle g, \boldsymbol{xy} \rangle$ を $\langle g', \boldsymbol{xy}' \rangle$ の後に追加する．$\boldsymbol{xy} = \boldsymbol{xy}'$ ならば，g' と g の小さい方をアイテム $\langle g', \boldsymbol{xy}' \rangle$ に代入する．(図 5.6 Step 9-18)

以上の処理を図 5.7を用いて具体的に説明する．アイテム (a) の累積コストが枝刈しきい値θより小さいとすると，上記処理 1) により，新たなアイテム (b),(c),...,(d) が生成される．ハッチングで示されているように，アイテム (a) の $[(\overline{x}_2, \overline{y}_2), \ldots, (\overline{x}_N, \overline{y}_N)]$ が伝送され，そのままアイテム (b),(c),...,(d) の $[(x_1, y_1), \ldots, (x_{N-1}, y_{N-1})]$ となる．上記処理 2) により，(c) のように生成されたアイテムがすでに $S_{x,y}$ 上にあるアイテムと同じウェイク像 $\boldsymbol{xy}$ を持つ場合，累積コスト g だけが更新される．

このように第 (i, j) 段の処理では，まず $LIST_{i,j-1}$の最大 $9R$個のアイテムに対して枝刈判定がなされ，その結果残った R個のアイテムから DP 漸化式処理が駆動されて最終的に $LIST_{i,j}$(すなわち $S_{x,y}$) が形成される．このビーム内データ駆動方式により，$\boldsymbol{XY}(i, j)$ すべてを走査することなく $LIST_{i,j}$を形成できるので，オーバーヘッドの大幅な低減が実現する．

このようなビーム内データ駆動方式を用いた場合，DP 漸化式の最小値選択処理をリスト上で行なうためには生成されたアイテムと同じ状態$\boldsymbol{xy}$ に関するアイテムが既にリスト S_{x_N, y_N} 上に存在するかどうかを調べる必要がある[2]．これをリスト S_{x_N, y_N} の盲目的な探索により行なえば，生成されたアイテム 1 つあたり $O(NR)$ の処理が必要になる．しかし，実際は上記処理 2) のようにリスト S_{x_N, y_N} の最後尾のアイテムとの比較を 1 回行なうだけそのような探索は必要なく，アイテム 1 つあたりの時間計算量は$O(N)$ で済む．

この利点は，リスト上の全アイテムが前述の順序関係に従ってソートされていることに因る．今，リスト $LIST_{i,j-1}$ 中のアイテムはソートされているとすれば，同じ $\boldsymbol{xy}$ を生成するようなアイテム $\langle \overline{g}, \overline{\boldsymbol{xy}} \rangle$ は $LIST_{i,j-1}$ 上で隣接している．これは，そのようなアイテムのウェイク像は $(\overline{x}_1, \overline{y}_1)$ 以外同一だから，M進数表現した場合最初の 2 桁のみが異なるためである．それゆえ $\boldsymbol{xy}$ を持つアイテムが新たに生成された場合，以前に同じ$\boldsymbol{xy}$を持つアイテムが生成されたならば，それは S_{x_N, y_N} の最

[2]実際，カラムワイズビームサーチ DP アルゴリズムではこのような処理が行なわれている (図 4.5 Step 30)．

後尾の (最新の) アイテムであることが保証される．

最後に，リスト上のアイテムが常に自動的にソートされていることを帰納的に示す．今 $LIST_{i,j-1}$ 上のアイテムはソートされているとする．ここで，$LIST_{i,j-1}$に関する処理をリストの先頭から順に行なうと (図 5.6, Step 3)，ある $S_{x,y}$ 上のアイテムは共通する$\boldsymbol{xy}$ の最上位 2 桁を除いた残りの $2N-2$ 桁に関して再びソートされている．この $S_{x,y}$ をまとめて $LIST_{i,j}$ とするとき $S_{1,1}$ を先頭としてラスタスキャン順に $S_{M,M}$ までつなぎ合わせれば，結局 $LIST_{i,j}$ 上の全アイテムはソートされていることになる．以上より最初に $LIST_{1,N}$ 上のアイテムをソートしておけば，その後生成されるリストのアイテムは自動的にソートされることがわかる．

5.6 計算量の評価

5.6.1 空間計算量

ピクセルワイズビームサーチ DP アルゴリズムの空間計算量は動的リストの利用によりピクセルワイズ DP アルゴリズムに比べて非常に少なくて済む．各$\boldsymbol{xy}(i,j)$ について $|\overline{\boldsymbol{XY}}(\boldsymbol{xy}(i,j))| \leq 9$ が成り立つので，$LIST_{i,j}$中のアイテム数は高々$9R$である．1 つのアイテムの大きさは$O(N)$ なので，$LIST_{i,j}$に必要な空間計算量は$O(NR)$ となる．よってアルゴリズムの空間計算量として，段数が$O(N^2)$ であることから $O(N^3R)$ が得られる．このように，画像サイズに対して多項式オーダーとなり，ピクセルワイズ DP アルゴリズムが要する空間計算量$O(N^3 9^N)$ に比べて大幅に低減されている． なおバックトラックが不要の場合，空間計算量は$O(NR)$ となる．

5.6.2 時間計算量

5.5で述べたように，ピクセルワイズビームサーチ DP アルゴリズムでは次の工夫により時間計算量を低減している．

1. ビーム内データ駆動方式
2. リストの探索が不要な DP 漸化式の最小値選択処理の実現

これらの工夫により，ピクセルワイズビームサーチ DP アルゴリズムでは，リストの 1 つのアイテムにつき $O(N)$ の処理で済む．これは$\overline{\boldsymbol{xy}}$ から$\boldsymbol{xy}$の生成 (図 5.6, Step 6) と，$\boldsymbol{xy}$と S_{x_N,y_N}の最後尾アイテムの$\boldsymbol{xy}'$の同一性判定 (図 5.6, Step 12) に要する計算量オーダーである．

アイテムが全体で$O(N^2R)$ 個生成されることを考えれば，アルゴリズムの時間計算量として$O(N^3R)$ が得られる．これは空間計算量と同じく多項式オーダーであり，ピクセルワイズ DP アルゴリズムが要する時間計算量$O(N^3 9^N)$ に比べて大幅に低減されている．

5.6.3　計算量と近似精度の関係

アルゴリズムの計算量と近似精度の関係を見るために，**4.6.2**と同じ画像 ($N = M = 6$) 100 対を用いて，$D(\boldsymbol{A}, \boldsymbol{B})$ に関する平均近似誤差，および平均計算時間 (Sun Ultra 1 ; SPECint_92: 215, SPECfp_92: 303) を実測した．結果を図 5.8に示す．カラムワイズビームサーチ DP アルゴリズムの場合と同様，ビーム径 Rを減少させれば計算時間を単調に短縮でき，逆に Rを増加させれば近似精度を単調に向上できることがわかる．

5.7　まとめ

本章では，最適な単調連続 2 次元ワープを探索する第二の DP アルゴリズムを検討した．このアルゴリズムは各段で 1 画素のマッピングを決定する多段決定過程に基づくものである．多段決定過程の設計においては，2 次元的単調連続性に関連して N段前の決定を参照するという N次マルコフ性を扱うためにウェイクなる概念を導入し，これの画素単位の推移過程を最適化するものとしてピクセルワイズ DP アルゴリズムを提案した．多段決定過程の規模を見積もることにより，アルゴリズムの時間および空間計算量がそれぞれそれぞれ$O(N^3 9^N)$，$O(N^3 9^N)$ であることを示した．

次にビームサーチの適用による近似アルゴリズム，ピクセルワイズビームサーチ

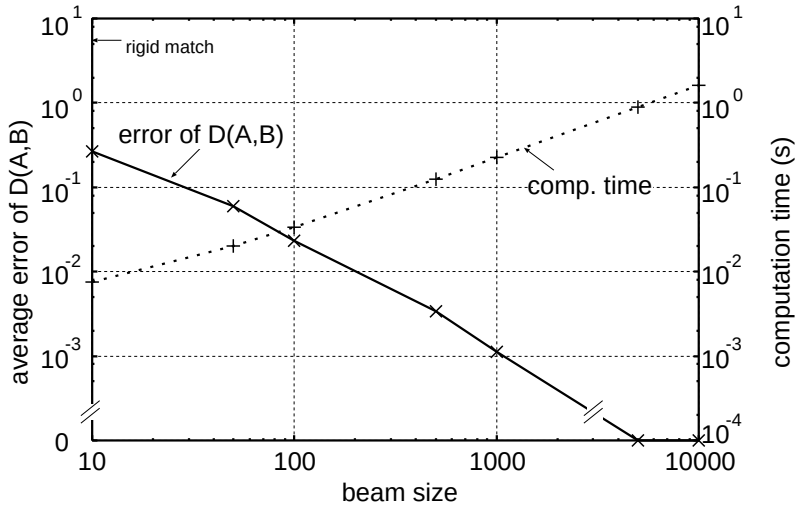


図 5.8: ピクセルワイズビームサーチ DP アルゴリズムの計算時間および近似誤差の平均値

DP アルゴリズムを提案した．カラムワイズ DP の場合とは異なり，ビームサーチの比較的素直な適用で多項式オーダーの計算量を持つ近似アルゴリズムが得られるが，ここではさらにリスト上での DP 漸化式計算に関する巧妙な工夫により，その時間計算量は$O(N^3R)$ で済むことを明らかにした．最後にこのアルゴリズムについても計算量と解の精度の間にトレードオフの関係があることを実験的に示した．

第 6 章
アルゴリズムの比較

6.1 まえがき

第 4章および第 5章で二種類の DP アルゴリズム，すなわちカラムワイズ DP アルゴリズムとピクセルワイズ DP アルゴリズムを提案した．これらの本質的な差異は，各段でのマッピングの決定単位にあり，前者が画像の各列を決定単位とするのに対し，後者は各画素を決定単位とする多段決定過程に基づくものであった．第 3 章で例示したように，基となる多段決定過程の定式化の違いにより，同じ問題を解く DP アルゴリズムでもそれらの計算量は大きく異なる場合がある．本章では以上二種類の DP アルゴリズムの計算量を比較検討する．

ビームサーチを適用した近似アルゴリズムについても，基となる探索空間の違いから，同じビーム径を用いたとしても得られる解の精度は異なると考えられる．またカラムワイズビームサーチ DP アルゴリズムでは，多項式オーダーの時間計算量とするために各部分遷移毎に枝刈を行なっており，この点も解の精度に影響すると考えられる．以上を明らかにするため，本章では両近似アルゴリズムの計算効率について理論的および実験的比較を行なう．

6.2 DP アルゴリズムの比較

カラムワイズ DP アルゴリズムとピクセルワイズ DP アルゴリズムは (2.2) 式の最小化問題に対し同じ最適解を与えるという意味で等価であるが，それらの計算量は大きく異なる．これまで述べたように，前者が$O(N^2 9^{2N})$ の時間計算量を要するのに対し，後者は$O(N^3 9^N)$ の時間計算量で済む．このことからピクセルワイズ DP アルゴリズムの方が少ない時間計算量で最適解を求めることがわかる．

この違いは多段決定過程の規模の違いに因る．特に 1 つの状態からの可能な遷

移数を比較すると，カラムワイズ DP アルゴリズムでは$O(9^N)$ と非常に多いのに対し，ピクセルワイズ DP アルゴリズムでは高々9 である．1 段当たりの状態数は共に$O(N9^N)$ であることからカラムワイズ DP アルゴリズムは二重に指数的な構造をしていると言える．このようにたとえ段の数がN倍であってもピクセルワイズ DP アルゴリズムの方が全体としての計算量は少なくなっている．

4.6.2，5.6.3と同じ乱数画像 100 対 $(N = M = 6)$ を用いて，最適ワープを求めるのに要する平均計算時間 (Sun Ultra 1 ; SPECint_92: 215, SPECfp_92: 303) を実測した．その結果，カラムワイズ DP アルゴリズムでは 42.1 秒，ピクセルワイズ DP アルゴリズムでは 4.37 秒であり，ピクセルワイズ DP アルゴリズムの方が 1 桁高速であった．この差はNの増加に従ってさらに大きくなるものと考えられる．

ところで，Levin らによる単調 2 次元ワープ [33] の DP アルゴリズムは，基本的にはカラムワイズ DP アルゴリズムから連続性条件を取り除いたものと等しい．その DP アルゴリズムを多段決定過程を用いて解釈すれば，段数はN，1 段の状態数は$O(N^{2N})$，1 つの状態から可能な遷移数は$O(N^{2N})$ となる．よって時間計算量は$O(N^{4N})$ であり，カラムワイズ DP アルゴリズムに比べ非常に多い．この主たる原因である状態数の増加 (図 4.4) は，連続性の欠如により取り得る局所ひずみパターンの数が増加するために起こる．このように連続性の導入は，自由度の点でワープの実用性を高めるだけでなく，計算量の大幅な低減にも貢献している．

6.3 ビームサーチ DP アルゴリズムの比較

6.3.1 近似精度の比較

カラムワイズ DP アルゴリズムにビームサーチを適用する時には，$\boldsymbol{xy}(i-1)$ から$\boldsymbol{xy}(i)$ への遷移を N回の部分遷移に分解して考え，その部分遷移毎に枝刈をする必要があった．これは単純にビームサーチを適用し，ある段で状態数を Rに絞っても，1 つの状態から$O(9^N)$ 通りの遷移が可能なため，次の段では一旦$O(9^N R)$ の状態が生成されてしまい，依然として指数オーダーのアルゴリズムしか得られないためであった．

このようにカラムワイズビームサーチDP アルゴリズムでは，結局ピクセルワイズビームサーチ DP アルゴリズムと同じく$O(N^2)$ 回分の枝刈処理を行なっている．しかし，後者が各段での枝刈終了後に DP 漸化式を実現するために同じ$\boldsymbol{xy}(i,j)$ に相当する探索経路のマージ (すなわち最小値選択) を行なっているのに対し，前者の部分遷移時にはそのようなマージは行われない．すなわちカラムワイズビームサーチ DP アルゴリズムでは，ピクセルワイズビームサーチ DP アルゴリズムの立場から見ればマージされるべき探索経路が同時に複数存在する可能性がある．よって両近似アルゴリズムで同じビーム径 Rを用いたとしても，カラムワイズビームサーチ DP アルゴリズムにおいて実効的な探索経路数は R以下であると考察される．

以上の考察を実験により確認した．まず，**4.6.2**と **5.6.3** で示した同じ $N = M = 6$ の乱数画像 100 対に対する近似誤差測定実験の結果を比較した．図 4.7および図 5.8 で示したビーム径に対する近似誤差の変化を表すグラフを，図 6.1に重ねて示す．また同実験において最適解に到達した割合を図 6.2に示す．これらの結果は共にすべてのビーム径 R についてピクセルワイズ DP アルゴリズムの方が平均的に近似精度が高いことを示している．なお，カラムワイズ DP アルゴリズムの場合，1 段当たりの状態数の最大値 $\max_i |\boldsymbol{XY}(i)|$ はおよそ 10000 である．しかし $R = 10000$ としても解は必ずしも最適解に一致していない．このことからもカラムワイズビームサーチ DP アルゴリズムの探索経路に冗長なものが含まれていることがわかる．

次に，画像のサイズとビーム径 Rを変えながら $D(\boldsymbol{A},\boldsymbol{B})$ を測定した．近似解は最適解未満になることはないので，この $D(\boldsymbol{A},\boldsymbol{B})$ が小さいほど近似精度が高いと判断できる．画像サイズとして $N = M = 8, 16, 32$ の 3 通りを用いた．各サイズについて，$[0, 1)$ の一様乱数で作成した画像 100 対を用意した．$D(\boldsymbol{A},\boldsymbol{B})$ の平均値を画素数 N^2で正規化した値を図 6.3 に示す．この結果からも，同じビーム径 Rで比べれば探索経路に冗長性のないピクセルワイズビームサーチ DP アルゴリズムの方が値が小さく，近似精度が高いことが確認された．

6.3.2　計算量の比較

2 つの近似アルゴリズムの時間計算量はそれぞれ$O(N^3R)$ および$O(N^2R^2)$ であ

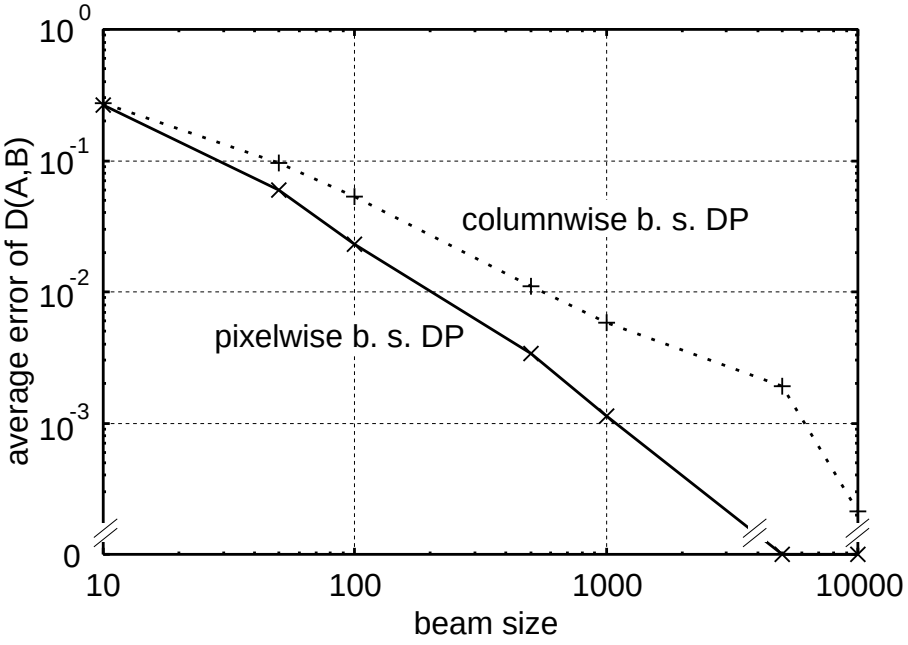


図 6.1: 近似アルゴリズムによる目的関数値 $D(\boldsymbol{A}, \boldsymbol{B})$ の最適解からのずれ

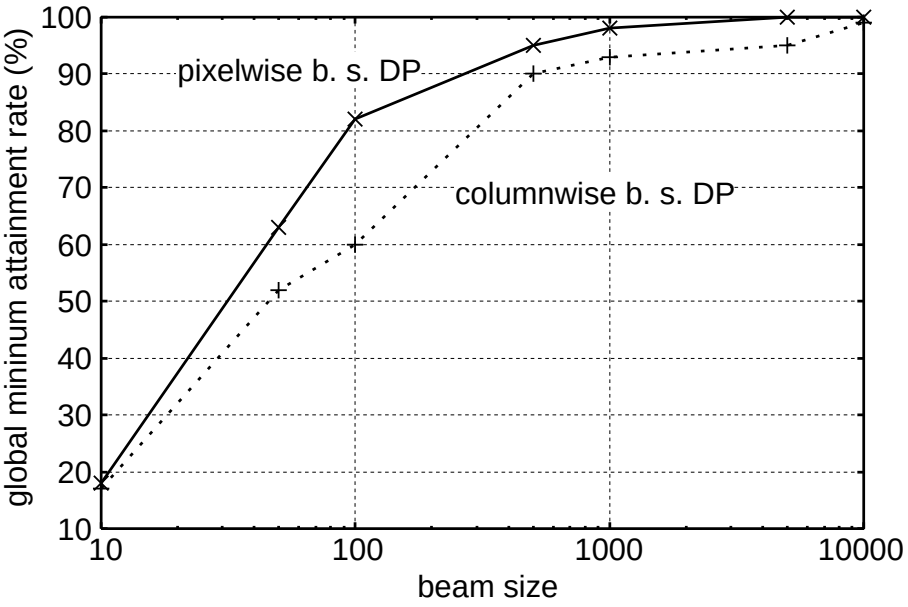


図 6.2: 最適解が得られた割合

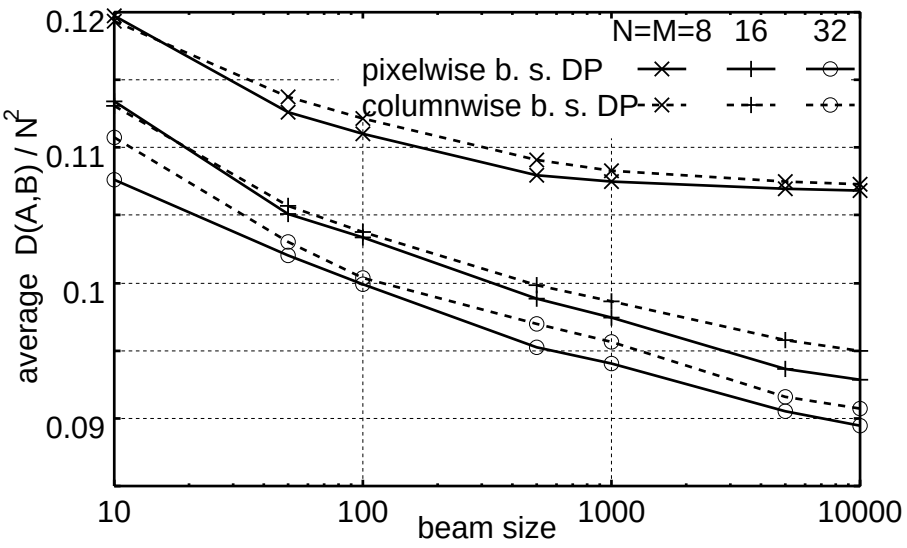


図 6.3: 近似アルゴリズムによる目的関数値 $D(\boldsymbol{A}, \boldsymbol{B})/N^2$

るため，これらの比較では一方のアルゴリズムの計算量的な優位性を結論することはできない．しかし，以下の実験結果が示すように，ピクセルワイズビームサーチ DP アルゴリズムの方が計算量効率がよい．

図 6.4 に Sun Ultra 2 (SPECint_95: 12.3, SPECfp_95: 20.2) を用いた時の，両近似アルゴリズムの計算時間の実測値を示す．このグラフを用いて同じビーム径 Rの時の計算時間を直接比較することはあまり意味がない．これは，前節で述べたように両アルゴリズムは同じビーム径 R を用いても近似精度が異なるためである．そこで同じ近似精度を与える時の計算時間を比較した結果を図 6.5に示す．これは図 6.3 と 図 6.4より求めた，同じワープ精度 $D(\boldsymbol{A}, \boldsymbol{B})/N^2$ を与えるときの計算時間の比の値である．例えば，この比の値が 0.5 であるとき，ピクセルワイズビームサーチ DP アルゴリズムの方が 2 倍速いことを意味する．この結果より，より高いワープ精度が必要な時ほど，ピクセルワイズビームサーチ DP アルゴリズムの方が効率がよく，最大で約 10 倍の高速化が図れることがわかった．

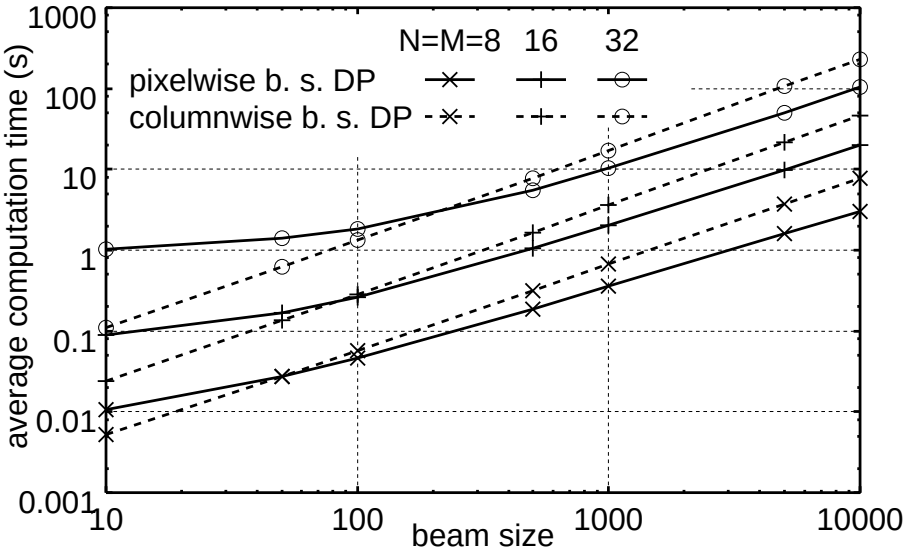


図 6.4: 近似アルゴリズムの計算時間

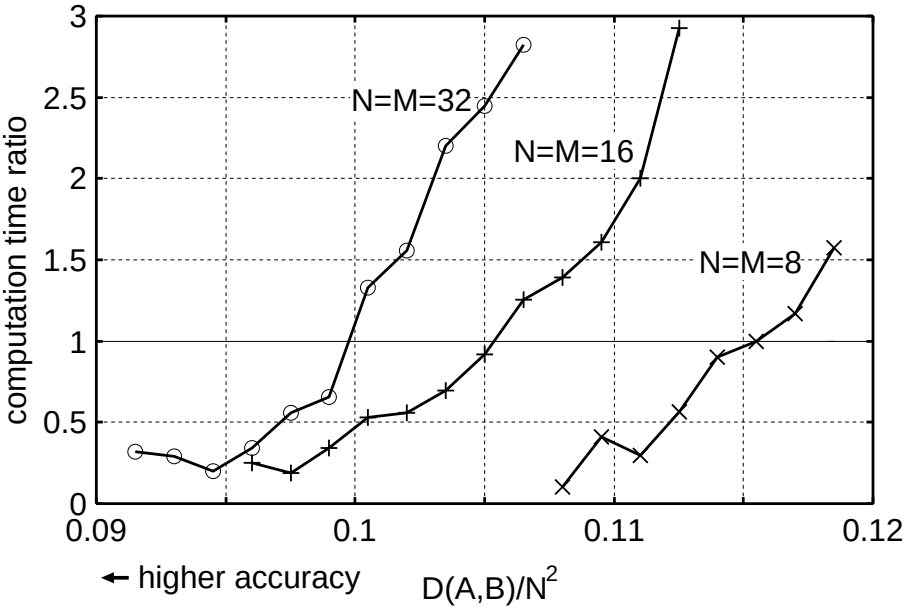


図 6.5: 2 つの近似アルゴリズムの計算時間の比

6.4 まとめ

本章では，まずカラムワイズ DP アルゴリズムとピクセルワイズ DP アルゴリズムの計算量の比較を行なった．その結果，同じ最適ワープを求める能力を有しながら，後者の方が大幅に少ない計算量で済むことが明らかになった．

次にそれらの近似アルゴリズムについても近似精度と計算時間に関して比較を行なった．同じビーム径 Rの時に得られるワープの精度を比較したところ，探索経路に冗長性の無いピクセルワイズビームサーチ DP アルゴリズムの方が高い精度を与えることがわかった．関連して同じ精度のワープを得るための計算時間を比較したところ，ピクセルワイズビームサーチ DP アルゴリズムの方が最大で 1 桁程度高速であることを確認した．

第 7 章
不自然なワープの回避

7.1 まえがき

これまでに述べたアルゴリズムをそのまま濃淡画像に適用すると，マッピング先の集中や全体としての滑らかさの欠如など，不自然な 2 次元ワープが生じる場合がある．後に示される図 8.1はその例である．この不自然なワープの原因としては，1) ビームサーチによる最適解からのずれ (近似誤差)，2) 単調連続性制約条件 (2.3)∼(2.6) の局所性，3) 画素特徴の不十分さによるマッピングの曖昧性，が考えられる．

以下，本章では，これら 1) と 2) に対処するために，ペナルティと整合窓について考察する．3) に関しては画素特徴量の高次元化が考えられるが，有効な特徴の選定は対象の性質に依存した問題となるため，ここでは議論しない[1]．

7.2 ペナルティ

本節では，ワープによる変形の量や質を定量的に評価する関数を本来の目的関数 (2.1) に加えることにより，制約条件により自由度を直接制限することなくワープによる変形量を制御する方法を考える．以下，この新たに付加される関数のことをペナルティ関数と呼ぶ．ペナルティ関数も目的関数と同様にワープ関数を変関数とする汎関数であり，正則化理論 [14] では類似のものを安定化汎関数と呼んでいる．

式 (2.1) の目的関数 Dにペナルティ関数 Pを加えた関数 $(D+P)$ を新たに目的関数として最小化を行なった場合，Dそのものを用いる場合に比べてそのペナルティ関数 Pを小さくするような 2 次元ワープが求まり易くなる．よって，もしペナルティ関数を望むべきワープの性質にうまく合致できれば不自然なワープを回避すること

[1] なお，**8.4**では画素特徴として 5 次元ベクトルを用いた文字認識実験について述べる．

ができる．さらにペナルティはビームサーチを用いた近似アルゴリズムにおいて枝刈に対する頑強性を高めることができる．これは $(D+P)$ を用いた場合，Pを小さくするような (望ましい)2 次元ワープとそうでない 2 次元ワープの目的関数値の差が Dだけを用いる場合に比べて一般に大きくなるためである．

ペナルティのもう一つの効果として，最適解に一意性を与える点が挙げられる．例えば，完全に一様な輝度値を持つ画像対を考えると，その間の最適 2 次元ワープは無数に存在する[2]．仮にこのような画像対に対しビームサーチを用いたアルゴリズムを適用すると，すべての探索経路が同じ累積コストを持つため，通常のしきい値処理ではすべての探索経路が通過することによるメモリオーバーフロー，もしくはすべての探索経路が枝刈される事態が生じる．2 値画像を対象とする場合はこれほど極端ではないが，同じ累積コストを持つ探索経路が数多く存在するので，やはり単純な枝刈処理では不具合が生じることが多い．これに対し，ペナルティ関数を導入すれば，累積コストのダイナミックレンジを増加できるので，画素特徴の改善や枝刈のしきい値処理に特別な工夫をしなくても，この問題をある程度回避することができる．

以上のようにペナルティの導入により，その最適化問題を解くにあたりデータの質やアルゴリズムの能力が不十分であっても，実用上十分な解が期待できる．

7.2.1 一様性ペナルティ

不自然なワープを防ぐためのペナルティ関数として，ワープ関数による画素の変位量を評価する関数の適用を検討する．これは，画像$\boldsymbol{A}$上のパターンと$\boldsymbol{B}$上のパターンの大局的構造は似ており，そのため最適なワープによる変形量は少ないという仮定に基づいている．この目的に合うペナルティ関数として，ここではワープ関

[2]オプティカルフローを求める時も，輝度値情報だけでは解が一意に定まらないという同様の現象が生じる．これは開口問題 (aperture problem) と呼ばれている [59]．この問題に対し，一般には「滑らかさ拘束」と呼ばれる動きベクトル場の 2 次偏微分値に基づく安定化汎関数が適用される [60]．

数の 1 次差分値の総和を用いる．

$$P_1 = \sum_{i=1}^{N}\sum_{j=1}^{N}\left(|p^x_{i,j}| + |q^x_{i,j} - 1| + |p^y_{i,j} - 1| + |q^y_{i,j}|\right) \tag{7.1}$$

ここで，

$$\begin{aligned} p^x_{i,j} &= x(i,j) - x(i,j-1), \qquad & q^x_{i,j} &= x(i,j) - x(i-1,j) \\ p^y_{i,j} &= y(i,j) - y(i,j-1), \qquad & q^y_{i,j} &= y(i,j) - y(i-1,j) \end{aligned}$$

ワープによる変形がない場合，このペナルティ関数 P_1の値は 0 になる．また平行移動に関しても 0 に近い値を返す．このようにペナルティ関数 P_1 はワープの一様性を評価している．

7.2.2 単調性ペナルティ

単調連続性条件 (2.3)∼(2.6) は，図 7.1に示すような部分的な反転を許容する．これは現在用いている単調性条件が，本来すべての方向について満たされるべき単調性の局所的かつ離散的近似であることに起因する．正確にゴム膜変換を模擬する 2 次元ワープを期待する場合，このような反転は望ましい結果を与えない．この反転を避ける方法として，4 点 (i,j)，$(i-1,j)$，$(i-1,j-1)$，$(i,j-1)$ がワープ後に構成する四辺形の面の向きを判定し，反転と判断された場合は以後の探索から除外する方法が考えられる．しかしビームサーチ DP アルゴリズムの場合，ビーム径 Rが小さいと残っている探索経路がこの判定ですべて除外されてしまう恐れがある．そこで反転の度合を反転部分の面積で表し，それに応じて次のペナルティP_2を課すことで反転を回避する．

$$\begin{aligned} P_2 \;=\; \sum_{i=2}^{N}\sum_{j=2}^{N} \Big\{ & \kappa(q^x_{i,j}p^y_{i,j} - q^y_{i,j}p^x_{i,j}) + \kappa(q^x_{i,j-1}p^y_{i,j} - q^y_{i,j-1}p^x_{i,j}) \\ & + \kappa(q^x_{i,j}p^y_{i-1,j} - q^y_{i,j}p^x_{i-1,j}) + \kappa(q^x_{i,j-1}p^y_{i-1,j} - q^y_{i,j-1}p^x_{i-1,j}) \Big\} \end{aligned} \tag{7.2}$$

ここで

$$\kappa(n) = \begin{cases} 0 & \text{if} \quad n \geq 0 \\ -n & \text{otherwise} \end{cases} \tag{7.3}$$

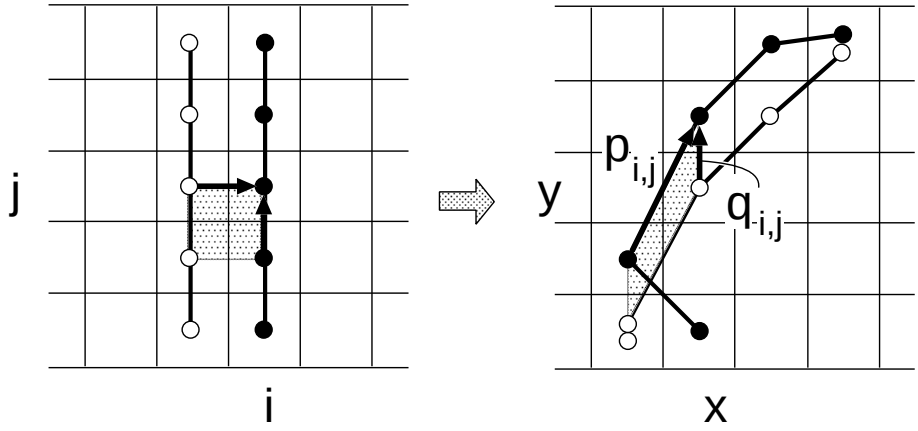


図 7.1: 単調連続性 (2.3)〜(2.6) を満たす反転の例

7.2.3 目的関数および DP 漸化式の変更

以上で定義したペナルティ関数 P_1，P_2 を (2.1) 式の Dに加えた新しい目的関数 D' は次のようになる．

$$D' = D + P = D + (\alpha P_1 + \beta P_2) \tag{7.4}$$

ここで，α，β はそれぞれペナルティ P_1，P_2 の効果を制御する重み (≥ 0) である．

カラムワイズ DP アルゴリズムの場合，この目的関数 D' に対する DP 漸化式は次のようになる．

$$g(i, \boldsymbol{xy}) = \min_{\overline{\boldsymbol{xy}} \in \overline{\boldsymbol{XY}}(\boldsymbol{xy})} \{d'(i, \boldsymbol{xy}, \overline{\boldsymbol{xy}}) + g(i-1, \overline{\boldsymbol{xy}})\} \tag{7.5}$$

ここで $\boldsymbol{xy} \in \boldsymbol{XY}(i)$ であり，

$$\begin{aligned} & d'(i, \boldsymbol{xy}(i), \boldsymbol{xy}(i-1)) \\ & \quad = \quad d(i, \boldsymbol{xy}(i)) + \alpha P_1(\boldsymbol{xy}(i), \boldsymbol{xy}(i-1)) + \beta P_2(\boldsymbol{xy}(i), \boldsymbol{xy}(i-1)) \end{aligned} \tag{7.6}$$

$$\begin{aligned} & P_1(\boldsymbol{xy}(i), \boldsymbol{xy}(i-1)) \\ & \quad = \quad \sum_{j=1}^{N} \left(|p_{i,j}^x| + |q_{i,j}^x - 1| + |p_{i,j}^y - 1| + |q_{i,j}^y| \right) \end{aligned} \tag{7.7}$$

$$
\begin{aligned}
&P_2(\boldsymbol{xy}(i), \boldsymbol{xy}(i-1)) \\
&\quad = \sum_{j=2}^{N} \Big\{ \kappa(q^x_{i,j} p^y_{i,j} - q^y_{i,j} p^x_{i,j}) + \kappa(q^x_{i,j-1} p^y_{i,j} - q^y_{i,j-1} p^x_{i,j}) \\
&\qquad + \kappa(q^x_{i,j} p^y_{i-1,j} - q^y_{i,j} p^x_{i-1,j}) + \kappa(q^x_{i,j-1} p^y_{i-1,j} - q^y_{i,j-1} p^x_{i-1,j}) \Big\} \qquad (7.8)
\end{aligned}
$$

一方，ピクセルワイズ DP アルゴリズムの場合，目的関数 D' に対する DP 漸化式は次のようになる．

$$
g(i, j, \boldsymbol{xy}) = \min_{\overline{\boldsymbol{xy}} \in \overline{\boldsymbol{XY}}(\boldsymbol{xy})} \{ d'(i, j, \boldsymbol{xy}, \overline{\boldsymbol{xy}}) + g(i, j-1, \overline{\boldsymbol{xy}}) \} \qquad (7.9)
$$

ここで $\boldsymbol{xy} \in \boldsymbol{XY}(i, j)$ であり，

$$
\begin{aligned}
&d'(i, j, \boldsymbol{xy}(i, j), \boldsymbol{xy}(i, j-1)) \\
&\quad = d(i, j, \boldsymbol{xy}(i, j)) + \alpha P_1(\boldsymbol{xy}(i, j), \boldsymbol{xy}(i, j-1)) \\
&\qquad + \beta P_2(\boldsymbol{xy}(i, j), \boldsymbol{xy}(i, j-1)) \qquad (7.10) \\
&P_1(\boldsymbol{xy}(i, j), \boldsymbol{xy}(i, j-1)) \\
&\quad = |p^x_{i,j}| + |q^x_{i,j} - 1| + |p^y_{i,j} - 1| + |q^y_{i,j}| \qquad (7.11) \\
&P_2(\boldsymbol{xy}(i, j), \boldsymbol{xy}(i, j-1)) \\
&\quad = \kappa(q^x_{i,j} p^y_{i,j} - q^y_{i,j} p^x_{i,j}) + \kappa(q^x_{i,j-1} p^y_{i,j} - q^y_{i,j-1} p^x_{i,j}) \\
&\qquad + \kappa(q^x_{i,j} p^y_{i-1,j} - q^y_{i,j} p^x_{i-1,j}) + \kappa(q^x_{i,j-1} p^y_{i-1,j} - q^y_{i,j-1} p^x_{i-1,j}) \qquad (7.12)
\end{aligned}
$$

以上のように，いずれの DP アルゴリズムでも遷移コスト d を d' に置き換える程度の修正でよく，ペナルティの導入による計算量の増加はほとんどない．実際，計算量のオーダーの変化は無い．この $d \to d'$ の置き換えを行うことで，近似アルゴリズムでも同様にペナルティを用いることができる．

7.3 整合窓

整合窓とは画素 (i, j) のマッピングを $(x, y) = (i, j)$ の近傍に制限する手法であり，ワープ関数に対する制約条件として定義される．$N = M$ の場合，w を非負の整

数として，その制約条件は次式で表される．

$$|x(i,j) - i| \leq w, \quad |y(i,j) - j| \leq w \tag{7.13}$$

整合窓の導入により，DP アルゴリズムの計算量は低減される．その第一の理由は，各段に属する状態がこの整合窓条件を満たすものに限られ，その結果漸化式計算回数が減少するためである．この効果は図 4.4からも類推できる．第二の理由は，整合窓条件によって除外された状態$\boldsymbol{xy}$にもともと遷移可能だった状態$\overline{\boldsymbol{xy}}$では漸化式内計算量が低減するためである．しかし，これら 2 点による計算量低減効果はさほど大きくなく，$w = 1$ の場合でもアルゴリズムの計算量は依然として指数オーダーに留まる．

一方，ビームサーチを用いた近似アルゴリズムでも，整合窓の導入により計算量が低減される．近似アルゴリズムの場合，状態数は Rで決定されるので漸化式計算回数の変化は無いが，幾つかの状態ではそこから駆動すべき状態数が減少する．ただしこの影響を受けるような状態がビーム内に存在しなければこの減少は無いので，実際の計算量低減効果はほとんどないと考えられる．

7.4 まとめ

本章では，単調連続性条件の不十分さやビームサーチの副作用により生じる不自然なワープを回避する方法として，ペナルティと整合窓を検討した．ペナルティに関しては，ワープによる変形量を制御するための一様性ペナルティと，単調連続性条件の局所性に起因した反転を回避するための単調性ペナルティを提案した．いずれも DP 漸化式の簡単な修正で導入でき，計算量の増加はほとんどないことを示した．

第 8 章

評価実験

8.1　まえがき

第 4章および第 5章では，それぞれの章で提案された近似アルゴリズムに対し，乱数画像を用いた最適化能力の定量的な評価実験を行なった．これらの実験の目的は純粋に近似精度の検証であり，2 次元ワープの変形追従能力や，単調連続性を制約条件とすることの意義は評価されていない．

本章では，単調連続 2 次元ワープの変形追従能力を定性的および定量的に評価するために幾つかの実験を行なった．最初に，画像間に実際に求めたワープの定性的な評価を行なった．特にペナルティ・整合窓の有無および連続性条件の有無がどのように結果に影響するかを考察する．次に最適 2 次元ワープが既知の画像対を用いて，ワープの変形追従能力を定量的に評価した．最後に手書きひらがなの認識実験により手書き変形に対するワープの追従能力を評価した．また，単調連続性の意義を定量的に評価するための比較対象として，各画素独立な摂動法による認識実験も行なった．

本章のすべての実験ではピクセルワイズビームサーチ DP アルゴリズムを用いた．これは，本章の実験で扱う画像のサイズでは現実的な時間で最適解が求まらないことと，6.3での考察によりカラムワイズビームサーチ DP アルゴリズムに対する優位性が確認されているためである．

8.2　単調連続 2 次元ワープの結果例とその定性的評価

8.2.1　ペナルティおよび整合窓の効果

ピクセルワイズビームサーチ DP アルゴリズムにより 6 組の画像間に実際に

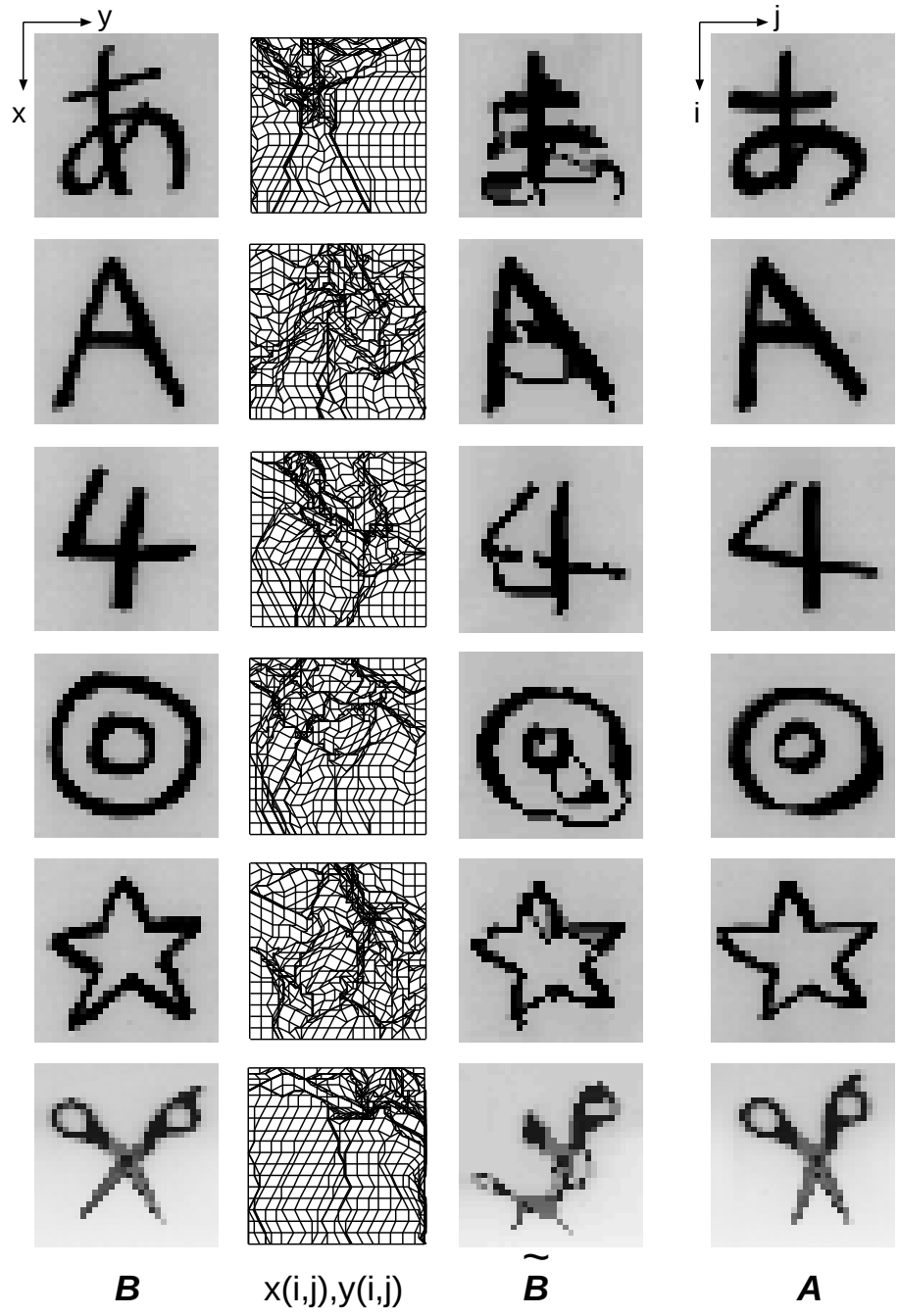


図 8.1: 単調連続 2 次元ワープの結果例 (ペナルティなし，整合窓なし)

求めた単調連続 2 次元ワープを図 8.1に示す．図の各行は左より，$\boldsymbol{B}$, ワープ関数 $(x(i,j), y(i,j))$ を表す変形格子，$\widetilde{\boldsymbol{B}}$，$\boldsymbol{A}$ である．用いた画像のサイズは $N = M = 32$，階調数は 256 であり，いずれもほぼ一様な輝度値を持つ背景上に線図形に近いパターンが記述されている．対とした$\boldsymbol{A}$，$\boldsymbol{B}$には同一カテゴリと見なせる程度の類似したパターンが記述されているので，ワープの結果は $\boldsymbol{A} \sim \widetilde{\boldsymbol{B}}$ となることが望ましい．ビーム径 Rは 1000 とした．この時の平均計算時間は約 12 秒であった (Sun Ultra 2 ; SPECint_95:12.3, SPECfp_95:20.2)．なお，ペナルティと整合窓はいずれも用いていない．変形格子の様子より (2.3)〜(2.6) の単調連続性条件を満たした 2 次元ワープが実現されていることがわかる．しかし，**7.1**で述べたように，至るところで不自然なワープが起きている．特に 「あ」や 「はさみ」では，前半の段のワープが $x = 1 \sim 10$ 付近に集中し後半の段でそのつじつまを合わせるような現象が生じている．またワープの集中している付近では **7.2.2**で指摘した反転が連鎖的に起きている．更に輝度値の微小な変化に敏感に反応し，全体としてゆらぎの多い不安定なワープとなっている．

次にペナルティ$(\alpha = 10, \beta = 100)$ を適用した場合の結果を図 8.2に示す．画像 $\boldsymbol{A}$，$\boldsymbol{B}$，およびビーム径は図 8.1の時と同じとした．得られた$\widetilde{\boldsymbol{B}}$を見るとすべてにおいてほぼ$\boldsymbol{A}$に一致していることがわかり，大局的構造を保存しようとするペナルティ P_1の有効性が示されている．また変形格子の様子よりわかるように輝度値の微小な変動に不感となる効果もある．更にペナルティP_2の効果も加わって，細部での反転もほとんど見られなくなっている．以上の結果から，ペナルティの適用によりワープの精度が改善できることが確認された．

図 8.3に整合窓 $(w = 2)$ を適用した場合の結果を示す．変形格子の様子から図 8.1中で生じていたワープの集中による極端な変形が解消されていることがわかる．しかし 「はさみ」の$\widetilde{\boldsymbol{B}}$のように整合窓による制約が強過ぎてワープがパターンの変形に追従できていない例も見られる．逆に制約を $w = 5$ 程度まで緩めると図 8.1と同様の不自然なワープが発生することが確認されており，適当な wを設定することの困難性を示唆している．このことから整合窓が特に有効なのは，$\boldsymbol{A}$，$\boldsymbol{B}$間の変形量のおよその上限がわかっており，更にそれが小さい場合と考えられる．

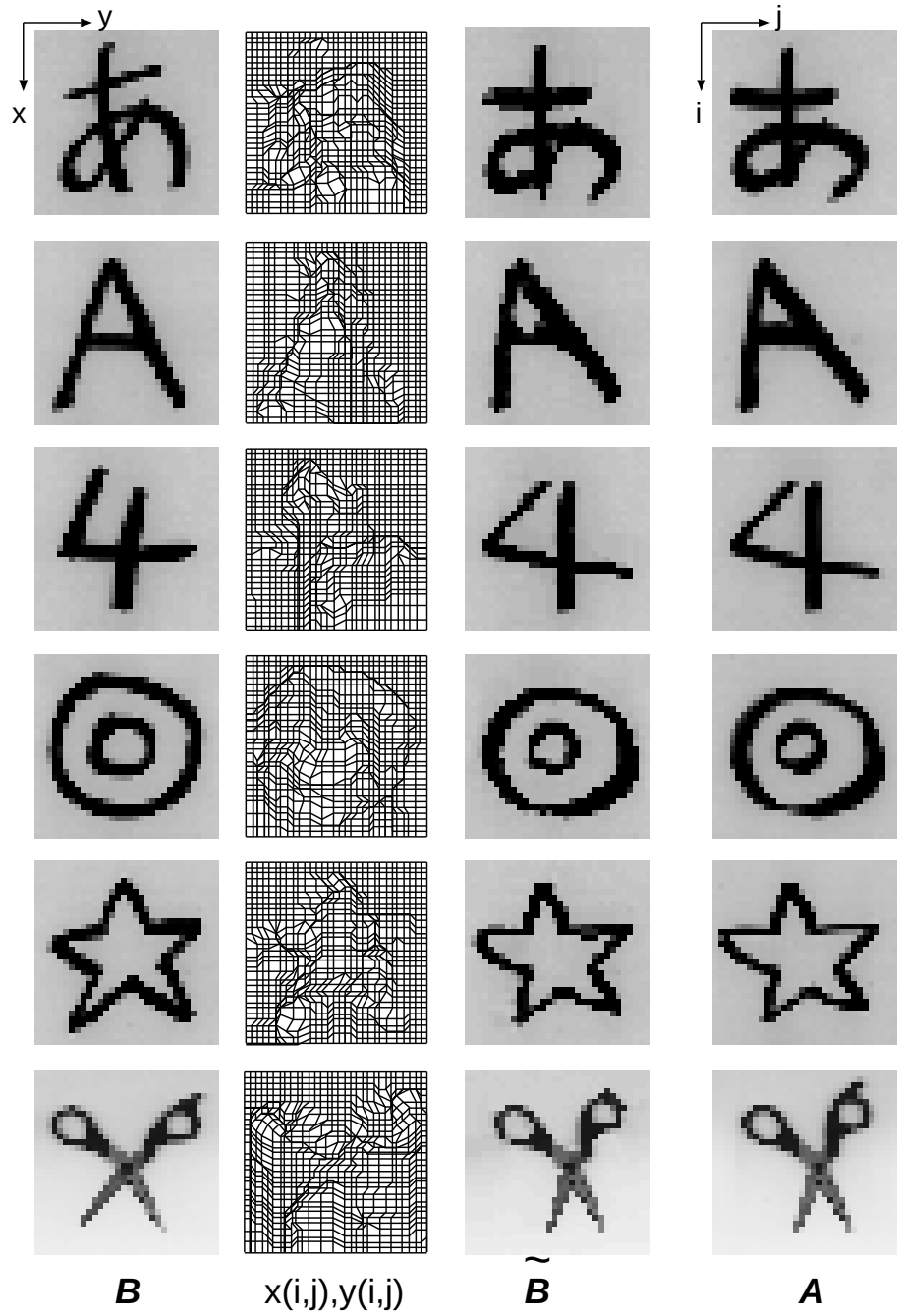


図 8.2: ペナルティ$(\alpha = 10, \beta = 100)$ 適用時の単調連続 2 次元ワープの例

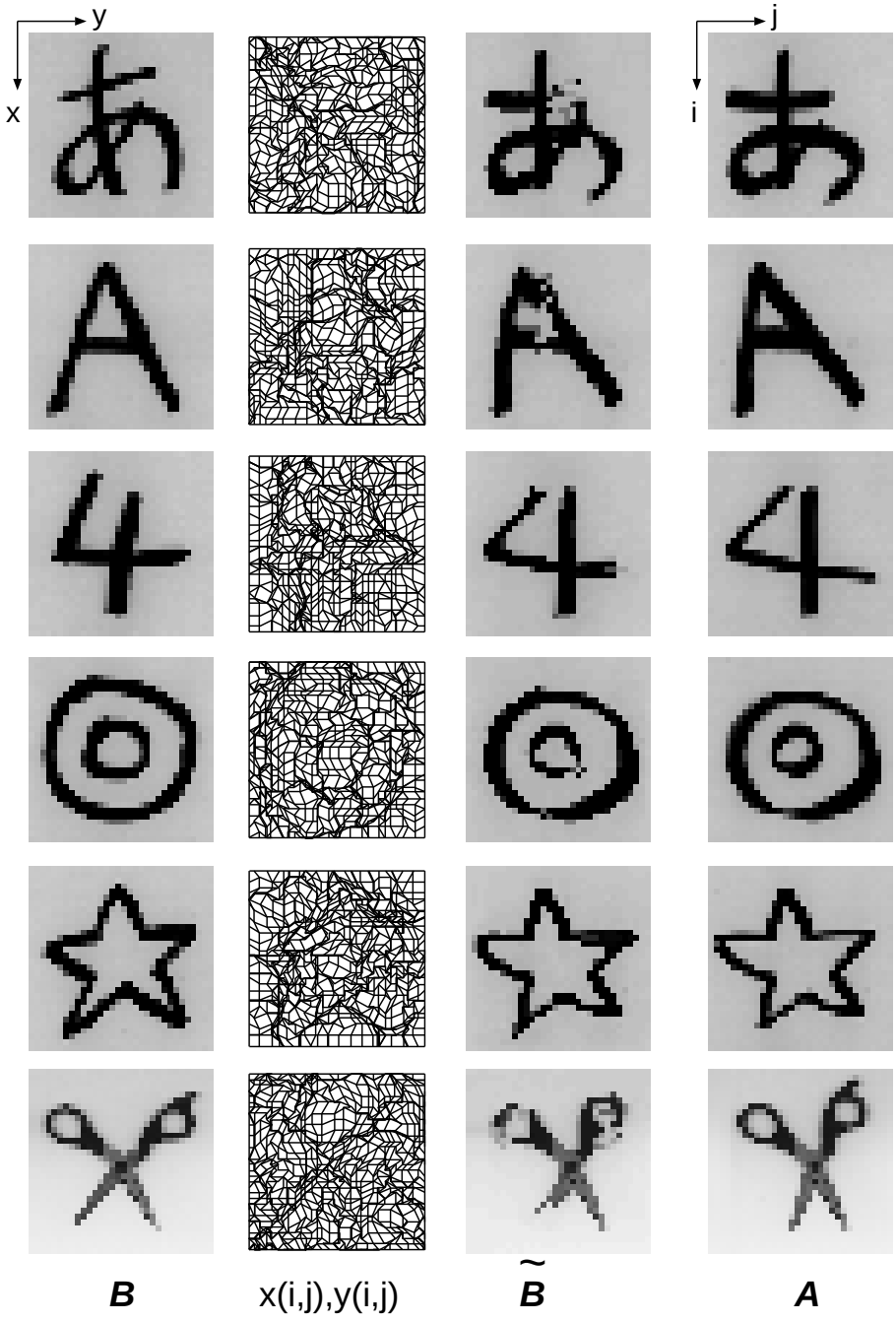


図 8.3: 整合窓 $(w = 2)$ 適用時の単調連続 2 次元ワープの例

8.2.2　連続性の効果

前節の実験で用いた画像対$\boldsymbol{A}$，$\boldsymbol{B}$は同一カテゴリに属すものと想定して用意したものであり，よってワープの結果としては$\boldsymbol{A} \sim \widetilde{\boldsymbol{B}}$を望んでいた．一方，単調連続 2 次元ワープをパターン認識問題に適用する場合，異なるカテゴリに属する$\boldsymbol{A}$と$\boldsymbol{B}$に関しては$\boldsymbol{A} \neq \widetilde{\boldsymbol{B}}$となることが望ましい．そこで，**8.2.1**で用いた画像$\boldsymbol{A}$，$\boldsymbol{B}$のすべての組合せについて，その間に単調連続 2 次元ワープを求める実験を行なった．結果を図 8.4に示す．ビーム径 $R = 1000$ とし，ペナルティおよび整合窓を併用した $(\alpha = 10,\ \beta = 100,\ w = 5)$．図中の数字は求まった $D(\boldsymbol{A}, \boldsymbol{B})$ であり，この値にペナルティ分 (P) は含まれていない．

同じカテゴリに属する画像間の画像間距離 $D(\boldsymbol{A}, \boldsymbol{B})$ と，異なるカテゴリに属する画像間の $D(\boldsymbol{A}, \boldsymbol{B})$ を比較すると，$2 \sim 8$ 倍程度後者の方が大きな値を示している．これは単調連続 2 次元ワープが過変形を起こしにくいことを示している．$\widetilde{\boldsymbol{B}}$の様子からもその傾向が確認できる．

このように過変形を避けることができた理由には，ペナルティや整合窓の効果もあると考えられるが，多くは連続性条件に因っていることが以下の実験より確認される．図 8.5に，これまでと同じ画像を用いた場合の，単調性のみを制約条件とする 2 次元ワープの結果を示す．ここでペナルティ$(\alpha = 10$，$\beta = 100)$ と整合窓 $(w = 5)$ に関しては図 8.4の場合と同一条件とした．用いたアルゴリズムはピクセルワイズビームサーチ DP アルゴリズムを基にしたものであるが，連続性の除外により $|\overline{\boldsymbol{XY}}(\boldsymbol{xy})|$ が$O(M^2)$ となりその計算量は増加する．この点を考慮して $R = 100$ とした．結果よりわかるように，全く異なるパターンでも$\boldsymbol{A} \sim \widetilde{\boldsymbol{B}}$となっているものが多く，ペナルティや整合窓を適用しているにもかかわらず過変形が確認される．$D(\boldsymbol{A}, \boldsymbol{B})$ の値を見ても図 8.4の場合のような大きな差は見られず，「☆」のように異なるカテゴリの方が小さな値を示している場合もある．

連続性の欠如は，以上のように過変形の原因となり，特にパターン認識問題においては致命的である．すなわち連続性は，**6.2**で述べたように計算量低減を実現するだけでなく，実用的な範囲の自由度を持つワープを与える点でも重要な条件と言える．

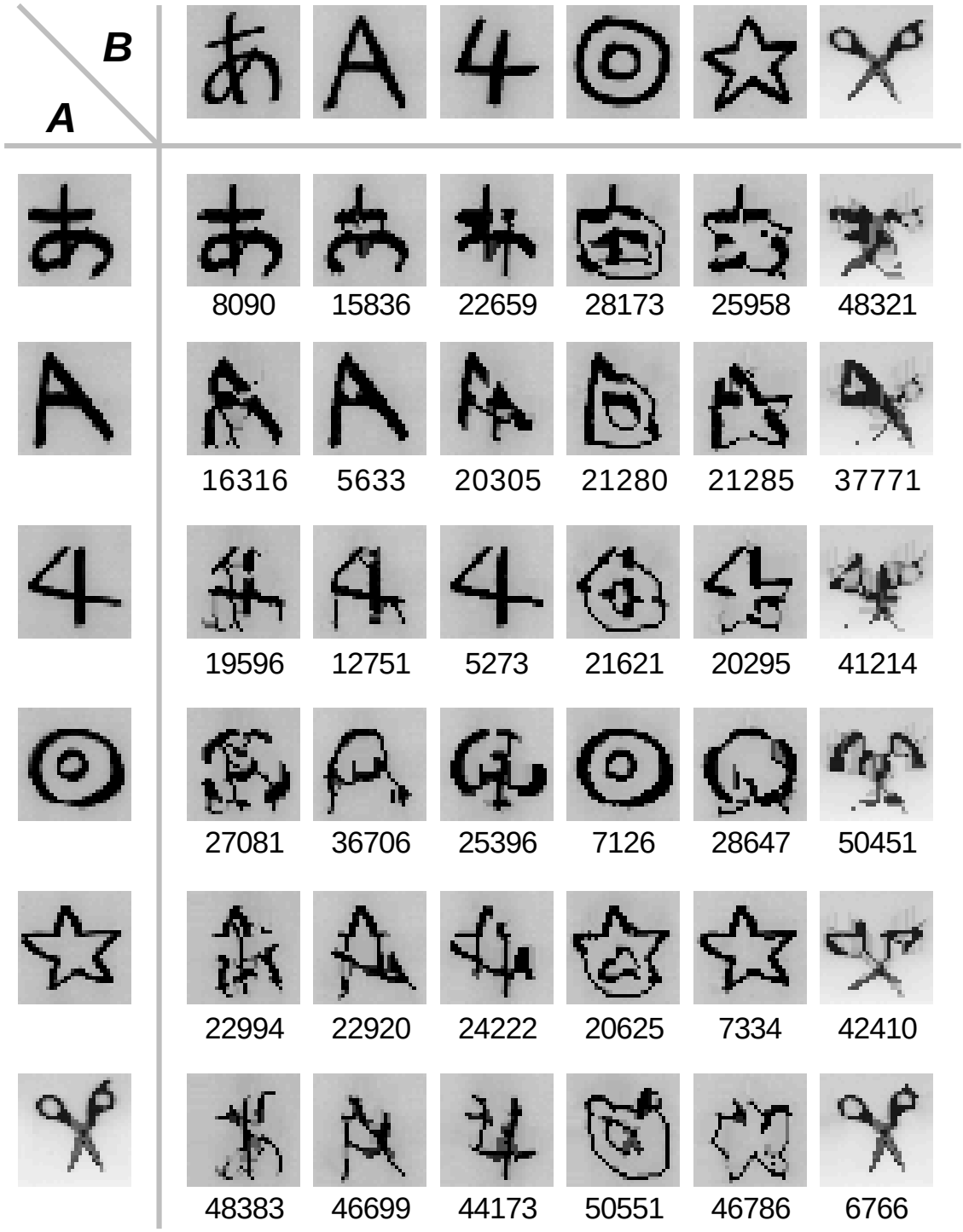


図 8.4: 単調連続 2 次元ワープによるワープ画像$\widetilde{\boldsymbol{B}}$と距離 $D(\boldsymbol{A}, \boldsymbol{B})$.

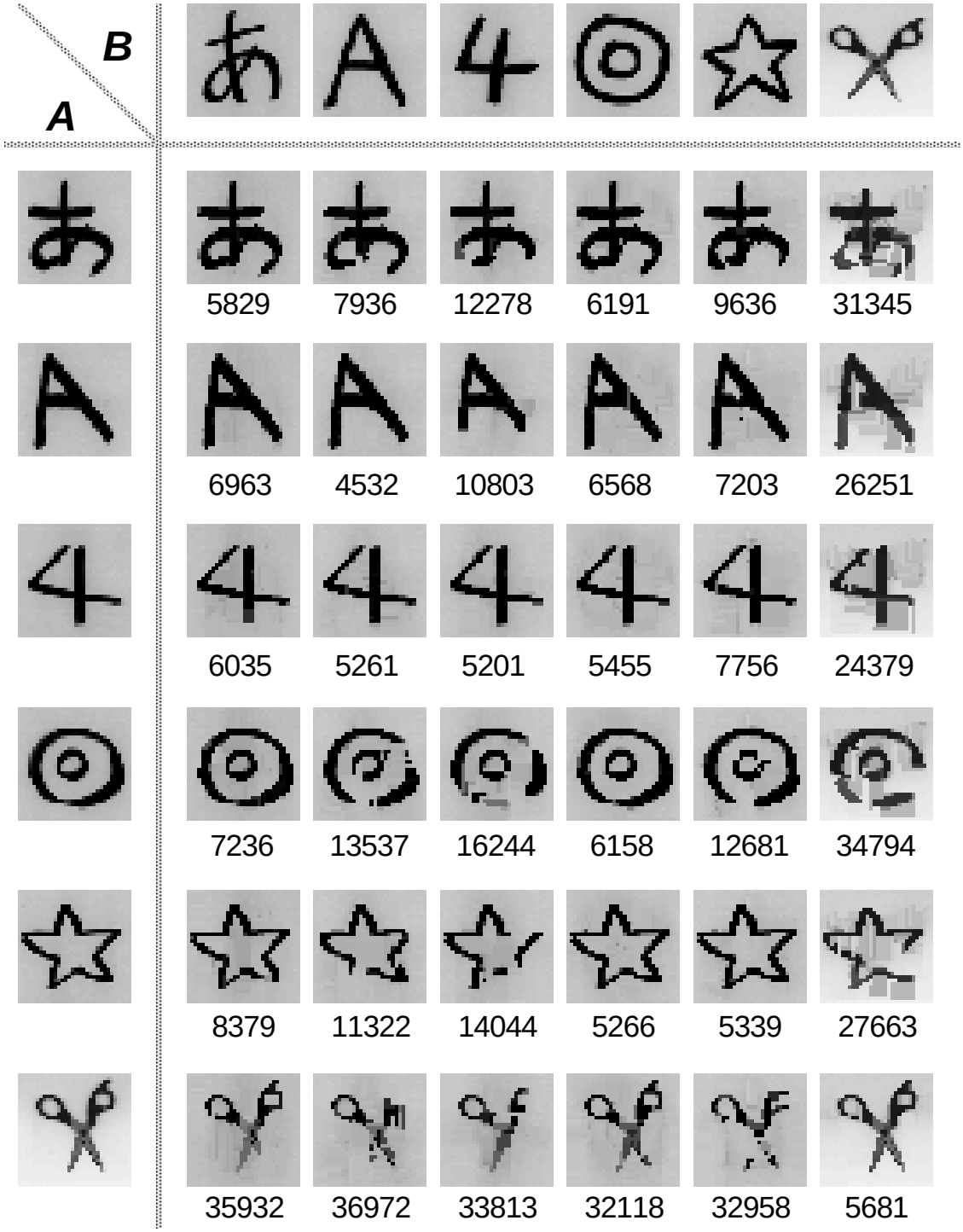
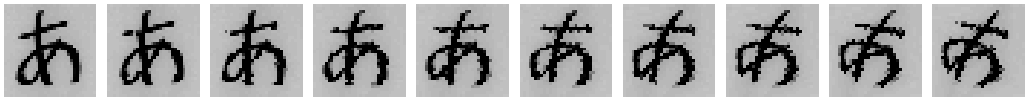


図 8.5: 単調 2 次元ワープによるワープ画像$\widetilde{\boldsymbol{B}}$と距離 $D(\boldsymbol{A}, \boldsymbol{B})$.

(a) translation



(b) rotation

(c) scaling

図 8.6: パラメトリックな変形により生成された画像$\boldsymbol{A}$の例

8.3 変形追従能力の定量的評価

単調連続 2 次元ワープの変形追従能力を定量的に評価する実験を行なった．ここでは最適 2 次元ワープが既知である$\boldsymbol{A}$，$\boldsymbol{B}$を用い，アルゴリズムによって得られたワープと最適 2 次元ワープのずれを評価基準とした．このため画像$\boldsymbol{A}$として，画像$\boldsymbol{B}$をパラメトリックに変形して生成した画像 $\{b(\overline{x}(i,j),\overline{y}(i,j)) \mid i,j=1,\ldots,N\}$ を用いた．このとき $\overline{x}(i,j)$，$\overline{y}(i,j)$ がそのまま最適 2 次元ワープとなる，変形追従能力の評価基準としては，変位抽出誤差ε [18] を用いた．

$$\varepsilon = 1/N^2 \sum_{i=1}^{N}\sum_{j=1}^{N} \sqrt{\{x(i,j)-\overline{x}(i,j)\}^2 + \{y(i,j)-\overline{y}(i,j)\}^2} \tag{8.1}$$

変形関数$\overline{x}$，$\overline{y}$としては，以下のいずれかを用いた．

平行移動 $(0 \leq t \leq 10)$:

$$\overline{x}(i,j) = i - tf_i, \quad \overline{y}(i,j) = j - tf_j$$

回転 $(0° \leq \theta \leq 45°)$:

$$\overline{x}(i,j) = i - (i'\cos\theta - j'\sin\theta - i')f_i, \quad \overline{y}(i,j) = j - (i'\sin\theta + j'\cos\theta - j')f_j$$

拡大・縮小 $(0.5 \leq s \leq 1.5)$:

$$\overline{x}(i,j) = i - (i' - i'/s)f_i, \quad \overline{y}(i,j) = j - (j' - j'/s)f_j$$

ここで $i' = i - N/2$，$j' = j - N/2$，$f_i = \sin(i\pi/N)$，$f_j = \sin(j\pi/N)$ である．f_i，f_j は変形を画像の範囲内に収めるために用いた．これらの$\overline{x}$，$\overline{y}$は実数値となるため四捨五入により整数値化した．画像$\boldsymbol{B}$として前節の実験と同じ画像 6 サンプル $(\boldsymbol{B})$ を用いた．この$\boldsymbol{B}$を変形して生成した$\boldsymbol{A}$の例を図 8.6に示す．

今回用いた変形パラメータの範囲では，$\overline{x}$，$\overline{y}$は単調連続性 (2.3)〜(2.6) を満たしていた．すなわちすべての場合で $D(\boldsymbol{A},\boldsymbol{B}) = 0$ かつ $\varepsilon = 0$ となる最適 2 次元ワープが存在する．言い換えると単調連続 2 次元ワープはこのような変形に追従する能力を理論上持っている．

第一の実験として，異なるビーム径 Rの下で変位抽出誤差εの 6 画像対分の平均値を求めた．結果を図 8.7に示す．ここでペナルティと整合窓は用いていない．$R = 1000$ の時を見ると，$\theta \geq 40°$や $s = 0.5$ 程度の大きな変形を除いては，εは 0.2 画素程度であり，単調連続 2 次元ワープの基本的な変形追従能力が定量的にも確認できた．またこの実験から R に対するε の単調減少性が認められる．すなわち R を大きくすれば平均的に変形追従能力が向上している．この傾向は **4.6.2**，**5.6.3**での乱数画像に対する $D(\boldsymbol{A},\boldsymbol{B})$ の測定実験の結果と一致している．なお，変形を全く行なわない場合でも $\varepsilon \neq 0$ となっているが，目的関数 $D(\boldsymbol{A},\boldsymbol{B})$ の値は 0 であった．

第二の実験として，ペナルティの重みαを変えながら同様にε を求めた．結果を図 8.8に示す．ここで $R = 1000$，$\beta = 100$ とした．整合窓は用いていない．$\alpha = 10000$ は単純重ね合わせに相当する．$\alpha = 1$ と$\alpha = 10$ の場合を比較すると，$\alpha = 10$ の方が変形追従能力が落ちることがわかる．これはペナルティP_1がワープによる変形量を少なくするためのエネルギーであることから説明される．

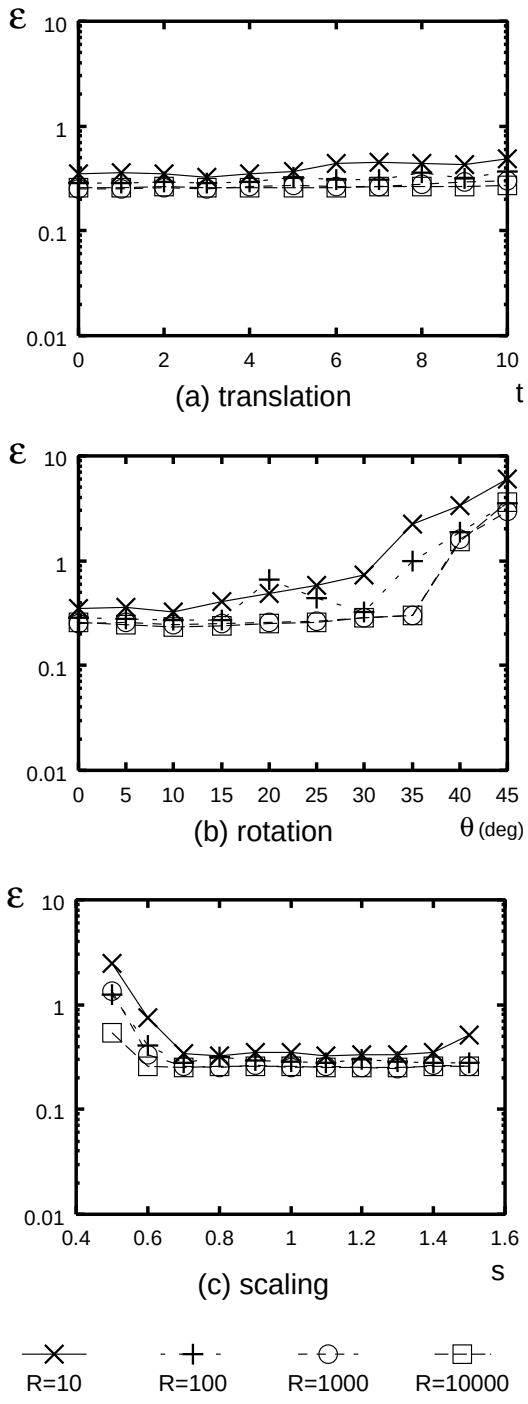


図 8.7: 変位抽出誤差 ε のビーム径 R に対する変化

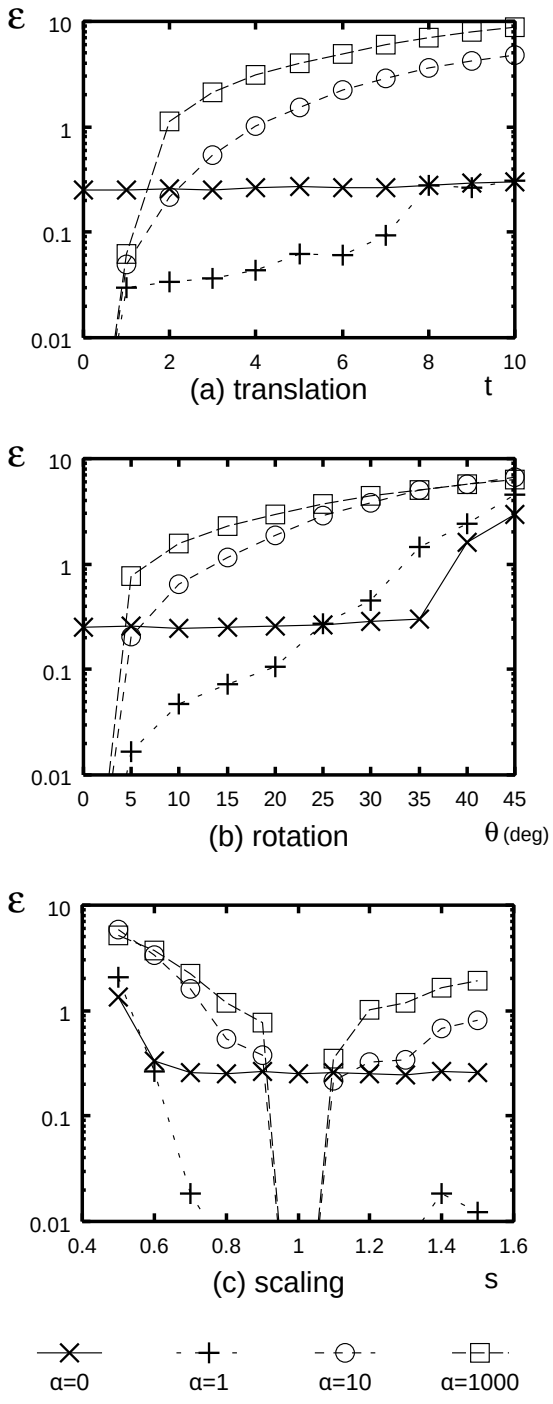


図 8.8: 変位抽出誤差 ε のペナルティ重み α に対する変化

8.4 手書きひらがな認識による評価

本節では，単調連続 2 次元ワープ法の変形追従能力の定量的評価を目的とした手書きひらがな (46 文字種) 認識実験の結果について述べる．

8.4.1 実験試料

実験試料として手書き文字データベース ETL8B(160 データセット) [61] を利用した．各文字パターンについて文字の大きさを正規化した後，「└ 」「\ 」「 |」「/ 」の 4 方向について線素方向特徴 [62] を抽出した．この結果，画素特徴は 1 画素あたり輝度値を含めて 5 次元ベクトルとなる．このように画素特徴を高次元化することで，ワープ精度の向上が期待できる．線素方向特徴の抽出後，縮小処理により画像サイズを $N = M = 16$ とした．以上の処理を図 8.9に示す．こうして得られた文字パターンのうち，偶数番 80 セットを入力パターン ($\boldsymbol{A}$，全 3680 文字) とした．残る奇数番の 80 セットについてはそれらを各文字種毎に単純平均して，各文字種につき 1 つの標準パターン ($\boldsymbol{B}$) を作成した．

8.4.2 画素特徴間距離

実験では画素特徴間距離として次式を用いた．

$$|a_I(i,j) - b_I(x,y)| + \eta \sum_{k=1}^{4} |a_D^k(i,j) - b_D^k(x,y)| \tag{8.2}$$

ここで $a_I(i,j)$，$b_I(x,y)$ はそれぞれ画素特徴 $a(i,j)$ と画素特徴 $b(x,y)$ の輝度値成分であり，$a_D^k(i,j)$，$b_D^k(x,y)$ $(k = 1,2,3,4)$ はそれらの線素方向特徴量成分である．$\eta(\geq 0)$ は線素方向特徴の重みを表す．

8.4.3 単純重ね合わせ法による認識実験

まず単純重ね合わせ法による認識実験を行った．認識は，パターン間距離

$$\sum_{i=1}^{N}\sum_{j=1}^{N}\left\{|a_I(i,j) - b_I(i,j)| + \eta \sum_{k=1}^{4} |a_D^k(i,j) - b_D^k(i,j)|\right\} \tag{8.3}$$

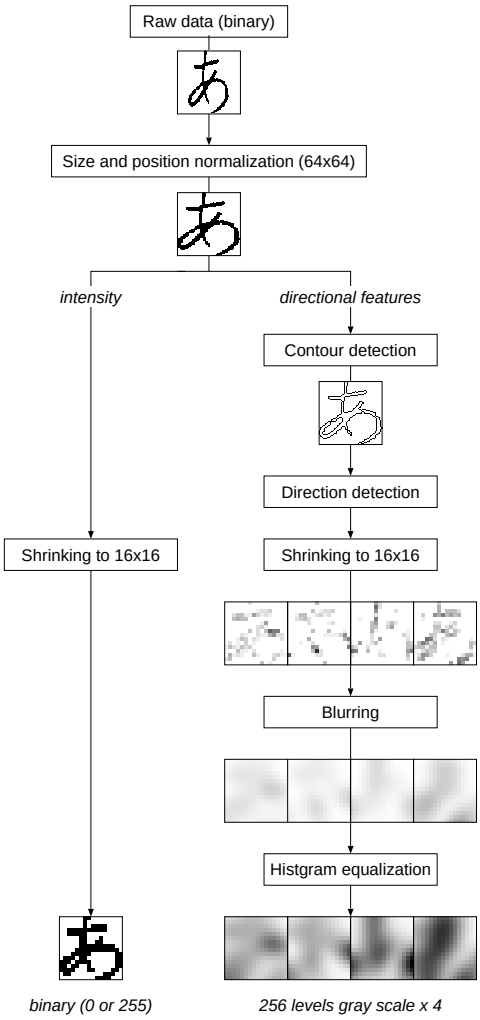


図 8.9: 文字パターンの前処理

を用いた最小距離識別法に依った．線素方向特徴の重み η を変えながら認識率を求めた結果を図 8.10に示す．これから $\eta = 0.5$ の時に最高認識率 93.9%が得られたことがわかる．

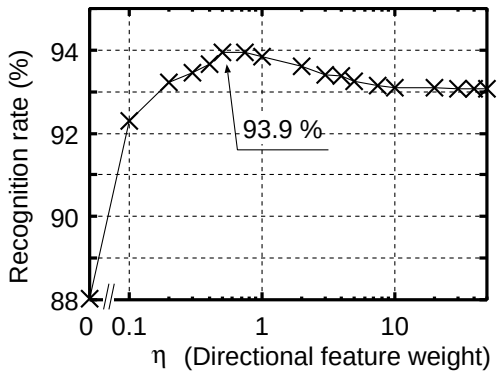


図 8.10: 単純重ね合わせ法による認識率

8.4.4 単調連続 2 次元ワープ法による認識実験

次に単調連続 2 次元ワープ法を用いて認識実験を行なった．目的関数 (2.1) は，(8.2) に合わせて変更した．ビーム径 Rは計算時間の都合上 1000 とした．このとき一対の文字パターン間にワープを求めるための計算時間はおよそ 3 秒であった (Sun Ultra 2 ; SPECint_95: 12.3,SPECfp_95: 20.2)．前節の結果より線素方向特徴の重み η は 0.5 とした．ペナルティと整合窓の効果を確かめるために，ペナルティを用いた場合と用いない場合の両方について，それぞれ整合窓幅 w を変えながら実験を行なった．なお，ペナルティの重み $(\alpha = 20, \beta = 100)$ は予備実験により決定した．

結果を表 8.1に示す．表中の結果 1 とは，目的関数値 (ペナルティ分を含む) をそのままパターン間距離として認識を行なった時の認識率である．結果 2 は，その目的関数値をそのまま用いずに，ペナルティの重み α と線素方向特徴の重み η を再設定してパターン間距離を計算した時の最高認識率である．よって結果 2 の認識率は結果 1 の認識率よりも高くなる．表 8.1より，結果 1 で最高 96.8%，結果 2 で最高 97.0%の認識率が得られたことがわかる．結果 1 でも単純重ね合せ法の結果と比べて 2.7%の向上であり，単調連続 2 次元ワープが手書き文字の変形をうまく吸収できていることを示している．

なお，最高認識率を与えた場合について，単調連続 2 次元ワープ法と単純重ね合

表 8.1: 単調連続 2 次元ワープによる手書きひらがなの認識率

ペナルティ		なし					あり $(\alpha = 20, \beta = 100)$				
整合窓幅 w		1	2	3	5	∞	1	2	3	5	∞
認識率	結果 1	94.8	96.0	95.9	95.2	94.5	95.6	96.5	96.8	96.3	96.2
	結果 2	95.5	96.4	96.2	95.4	94.7	96.2	96.9	97.0	96.4	96.3

わせ法で誤認識された文字パターンを比較すると，

- 単純重ね合わせで誤認識，単調連続 2 次元ワープで認識$\cdots$ 139 文字
- 単純重ね合わせで認識，単調連続 2 次元ワープで誤認識$\cdots$ 26 文字
- 単純重ね合わせ，単調連続 2 次元ワープ，共に誤認識$\cdots$ 84 文字

であった．

8.4.5 摂動法による認識実験

単調連続性の有効性を確認するための比較対象として，各画素独立な摂動法 (ずらし) [24] [1]による認識実験を行なった．最大ずらし幅を vとすれば，摂動法によるパターン間距離は

$$\sum_{i=1}^{N}\sum_{j=1}^{N}\left[\min_{\substack{|\ i-x(i,j)\ |\ \leq\ v\\ |\ j-y(i,j)\ |\ \leq\ v}}\left\{|\ a_I(i,j)-b_I(x(i,j),j(i,j))\ | + \eta\sum_{k=1}^{4}\left|\ a_D^k(i,j)-b_D^k(x(i,j),y(i,j))\ \right|\right\}\right] \tag{8.4}$$

と表される．ここでワープ関数には境界条件 (2.7) のみが制約条件として与えられる．単調連続性制約が無いので，上式は各 (i,j) 独立の最小化問題として解くこと

[1]アフィン変換のパラメータを少しずつ変化させて生成した変形標準パターンをあらかじめ数多くして用意しておくパターンマッチング手法[11]~[13] のことを摂動法 (perturbation method) と言う場合もあるが，ここでいう摂動法とはワープ関数に対する制約条件の無い 2 次元ワープ法のことを指す．

表 8.2: 摂動による手書きひらがなの認識率

最大ずらし幅 v	1	2	3
認識率	94.3	92.8	90.0

ができる．η は前節と同じく 0.5 とした．なお，摂動法では原理的にペナルティP_1，P_2を用いることはできない．

結果を表 8.2に示す．最大ずらし幅 $v = 1$ の時は単純重ね合わせ法よりも高い認識率 94.3%を得ているが，vの増加に対する認識率の低下が激しく，$v = 2$ の時にはすでに単純重ね合わせ法よりも低い認識率となっている．これらはずらし幅が大きくなったことによる変形吸収能力の向上よりも，過変形による性能劣化の方が大きいためと考えられる．

摂動法による認識率と，単調連続 2 次元ワープ法のペナルティを用いない場合の認識率 $(\text{結果}\ 1)$ を同じ w，vについて比べれば，厳密に単調連続性の効果だけを評価することができる．単調連続 2 次元ワープ法の結果 $(\text{表}\ 8.1)$ を参照すれば，$v = w = 1, 2, 3$ の場合にそれぞれ 0.9%，3.2%，5.9%ほど摂動法の方が低い認識率となっている．さらに単調連続 2 次元ワープ法では wの増加に対し比較的安定した認識率を示している．この結果から，単調連続性をワープの制約条件として用いることにより過変形を押えながら手書き文字の変形をうまく吸収できることが確認できた．

8.5 まとめ

本章では，単調連続 2 次元ワープの変形追従能力を定性的および定量的に評価するための実験とその結果について述べた．

最初に，画像間に実際に求めた 2 次元ワープを定性的に評価する実験を行なった．その結果より，ビームサーチの副作用や単調連続性条件の不十分さにより不自然なワープが生じる場合があることがわかった．また，この問題に対するペナルティや整合窓の有効性を確認した．さらに連続性制約が過変形を防ぐために不可欠である

ことを，単調性のみを制約条件とする 2 次元ワープとの比較実験により確認した．

次に，最適 2 次元ワープが既知の画像対を用いてワープの変形追従能力を定量的に評価する実験を行なった．その結果，平行移動，回転，拡大・縮小による変形をかなりの程度まで追従できることがわかった．

最後に，手書きひらがなの認識率により手書き変形に対するワープの追従能力を評価する実験を行なった．その結果より単調連続 2 次元ワープ法がパターン認識において適当な変形追従能力を有することが示唆された．また各画素独立にワープを行なう摂動法との比較により，単調連続性条件の有効性を確認した．

第 9 章

結論

本研究は単調連続性を制約条件とする 2 次元ワープ法に関して基礎的な検討を行なったものである．

まず，2 次元ワープを 2 画像間の最大一致を実現する画素間のマッピングを求める最適化問題として定式化した．ここで画像の位相構造を保存する 2 次元ワープを実現するための制約条件として単調連続性を定義した．次にこの単調連続 2 次元ワープの決定問題に対し，その最適解を与える動的計画法 (DP) に基づくアルゴリズムを 2 種類示した．DP に基づく 2 次元ワープ法はこれまでにも幾つかの手法が提案されているが，複雑さや計算量の多さを予想して，いずれも自由度もしくは最適性を犠牲にしたものや問題の本質を把握していない不十分なものであった．これに対し，本論文で提案した DP アルゴリズムは位相構造を保存する範囲での最大の自由度を持ち，かつ解の最適性を保証する正攻法的なアプローチに基づいたものであり，これらを出発点として様々な発展が望めるものと考えている．

カラムワイズ DP アルゴリズムでは画像中の全 N^2個の画素のマッピングの決定問題を画像の各列を処理単位とすることで N段の多段決定過程として取り扱っている．一方，ピクセルワイズ DP アルゴリズムでは同じ問題をラスタスキャンの順序に従う N^2段の多段決定過程として取り扱っている．この取り扱い方の違いにより，どちらのアルゴリズムも同じ最適解を与える能力を有しながら，ピクセルワイズ DP アルゴリズムの方が大幅に計算量が少なくて済むことを明らかにした．

実用を目指した改良として，それぞれの DP アルゴリズムについてビームサーチを適用した近似アルゴリズムを提案した．ビームサーチを DP アルゴリズムに組み込むことで最適解の保証はできなくなるが，ワープの 2 次元的自由度を保ったまま，時間および空間計算量を共に多項式オーダーに低減できた．基となる DP アルゴリズムの違いのため，これら近似アルゴリズムに効率の違いがあることを理論的および実験的に示した．すなわち，ピクセルワイズ DP アルゴリズムに基づく近似アル

ゴリズム (ピクセルワイズビームサーチ DP アルゴリズム) の方が探索空間に冗長性がないために実質的に少ない計算時間で同一精度の近似解を求めることを明らかにした．

最後に単調連続 2 次元ワープの変形追従能力を定量的および定性的に評価するために，幾つかの実験を行なった．それらの結果は次のようにまとめられる．

- ビームサーチの副作用などの原因で不自然なワープが生じる場合があるが，ペナルティや整合窓の適用により改善できる．
- 単調性に加えての連続性制約条件は過変形を防ぐために不可欠である．
- 平行移動，回転，拡大・縮小による変形や手書き文字に発生する変形をかなりの程度まで追従できる．

今後の課題として第一にアルゴリズムの計算量の一層の低減が挙げられる．この点については，本論文では扱わなかった対象固有の変形特性の利用を含めて検討する必要がある．

第二の課題として，単調連続 2 次元ワープの応用に関する検討が挙げられる．2 次元ワープをパターン認識問題へ適用する場合，対象に発生する様々な変形への追従能力を残したまま，いかに過変形を防ぐかが問題となる．この問題に対し，さまざまな制約条件や画素特徴が取り扱えるという DP アルゴリズムの特長を最大限に活用すれば，本手法を効果的に応用できると考えられる．

謝辞

本論文をまとめるまでに，多くの方々の御指導，御協力を賜わった．

九州大学迫江博昭教授には，多大なる御指導，御鞭撻を賜わった． ここに謹んで謝意を表します．全く新しいテーマとしてスタートした本研究を成功裡に終結し，学位論文としてまとめることができたのは氏の適切な御指導に負うところが大きい．長谷川勉教授，谷口倫一郎教授には本論文をまとめるにあたり有益な御助言を賜わった．深く感謝する次第である．鎌田清一郎助教授には，画像処理に関する多くの御指導のみならず，学会活動に関する御助言も頂いた．また，片山助手を始めとする研究室諸氏にも御支援を頂いた．ここに感謝したい．

セコム株式会社 IS 研究所の橋本新一郎所長，古村光夫 SE グループ主幹研究員 (元 ・音声情報処理研究室室長) には，平成 4 年度より 7 年度までの 4 年間，懇切なる御指導を賜わった．IS 研究所で行なった研究の面白さのあまり勢い余って大学に戻りたいと言いだした私を 「がんばって来なさい」とお許し下さった両氏に深く感謝したい．また，濱上知樹氏を始めとする音声情報処理研究室の方々にも心よりお礼申し上げたい．私の IS 研究所時代のテーマと本研究のテーマは直接結び付くものではないが，研究の方法論を始め IS 研究所で学んだ多くのことが本研究の礎となっていることは間違いない．

参考文献

[1] George Wolberg. *Digital image warping*. IEEE Computer Society Press, Los Alamitos, California, 1990.

[2] 茨木俊秀. 組合せ最適化の理論. 電子情報通信学会, 1979.

[3] T.H. Cormen, C.E. Leiserson, R.L. Rivest 著, 浅野, 岩野, 梅尾, 山下, 和田共訳. アルゴリズムイントロダクション. 近代科学社, 1995.

[4] Michael R. Garey and D. S. Johnson. *Computers and Intractability : A Guide to the Theory of NP-Completeness*. W H Freeman & Co, 1979.

[5] R. ベルマン, S. ドレイファス著, 小田中敏男, 有水　彊訳. 応用ダイナミック・プログラミング. 日本科学技術連盟・JUSE 出版, 1962.

[6] 鍋島一郎. 動的計画法, 数学ライブラリー, 第 7 巻. 森北出版, 1968.

[7] 近藤次郎. 最適化法, 現代応用数学講座, 第 6 巻. コロナ社, 1984.

[8] Bernard Widrow. The "rubber-mask" technique – Part I and II. *Pattern Recognition*, Vol. 5, pp. 175–211, 1973.

[9] Toru Wakahara. Shape matching using LAT and its application to handwritten numeral recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 16, No. 6, pp. 618–629, 1994.

[10] Anil K.Jain, Yu Zhong, and Sridhar Lakshmanan. Object matching using deformable templates. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 18, No. 3, pp. 267–278, 1996.

[11] Michio Yasuda, Kazuhiko Yamamoto, and Hiromitsu Yamada. Effect of the perturbed correlation method for optical character recognition. *Pattern Recognition*, Vol. 30, No. 8, pp. 1315–1320, 8 1997.

[12] Thien M. Ha and Horst Bunke. Off-line, handwritten numeral recognition by perturbation method. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 19, No. 5, pp. 535–539, 5 1997.

[13] 中山直樹, 春山 智, 坂野 鋭. 摂動法による顔画像の認識. 電子情報通信学会技術研究報告, PRMU98-90, 9 1998.

[14] 坂上勝彦, 横矢直和. 弛緩法と正則化. 情報処理, Vol. 30, No. 9, pp. 1047–1057, 1989.

[15] 横矢直和, 坂上勝彦. 画像理解と最適化原理. 電子情報通信学会誌, Vol. 74, No. 4, pp. 326–334, 1991.

[16] 山本和彦, 坂上勝彦, 積山洋子. Active Net によるステレオマッチング. 電子情報通信学会技術研究報告, PRU90-12, 1990.

[17] 横矢直和. 多重スケールでの正則化によるステレオ画像からの不連続を保存した曲面再構成. 電子情報通信学会論文誌, Vol. J76-D-II, No. 8, 1993.

[18] 水上嘉樹, 古賀和利, 鳥岡豊士. 変位抽出を行う手書き文字認識システム. 電子情報通信学会論文誌, Vol. J80-D-II, No. 1, pp. 63–72, 1997.

[19] ズデネク プロハースカ, 伊藤崇之, 岡本敏雄. 変分法に基づく画像変形. 電子情報通信学会技術研究報告, NC97-163, pp. 185–192, 3 1998.

[20] A.Daffertshofer and H.Haken. A new approach to recognition of deformed patterns. *Pattern Recognition*, Vol. 27, No. 12, pp. 1697–1705, 1994.

[21] 柳田　正, 永崎　健, 中川正樹. 弾性膜モデルに基づくオフライン手書き文字認識. 電子情報通信学会技術研究報告, PRMU97-223, 2 1998.

[22] 目黒眞一, 梅田三千雄. 階層的パターンマッチングによる手書き文字変形の抽出. 電子情報通信学会技術研究報告, PRL77-70, 1977.

[23] D.J.Burr. A dynamic model for image registration. *Computer Graphics and Image Processing*, Vol. 15, pp. 102–112, 1981.

[24] 山田博三, 斉藤泰一, 森　俊二. 類似度法の一改良 – ずらし類似度 –. 電子通信学会論文誌, Vol. J64-D, No. 10, pp. 970–976, 1981.

[25] 泉井良夫, 原島　博, 宮川　洋. 階層的な辞書の変形を用いた手書き文字認識. 電子通信学会論文誌, Vol. J68-D, No. 3, pp. 361–368, 1985.

[26] Yoshihisa Shinagawa and Toshiyasu L. Kunii. Unconstrained automatic image matching using multiresolutional critical-point filters. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 20, No. 9, pp. 994–1010, 9 1998.

[27] 中野康明, 中田和男, 中島　晃. 周辺分布とそのスペクトルによる漢字認識の改良. 電子通信学会論文誌, Vol. 57-D, No. 1, pp. 15–22, 1974.

[28] 塩野　充, 真田英彦, 手塚慶一. DP を用いた伸縮整合法による常用手書き文字の認識実験. 電子情報通信学会技術研究報告, PRL80-22, 1980.

[29] 田中直樹, 塩野　充, 真田英彦, 手塚慶一. 動的方向性整合法による手書き漢字認識. 電子通信学会論文誌, Vol. J68-D, No. 1, pp. 56–63, 1985.

[30] 磯道義典, 竹政富尚. 航空写真からの高度図の作製. 第 7 回画像工学コンファレンス, pp. 31–34, 1976.

[31] 松本　弘, 脇田　壽. Frequency warping による話者の正規化. 日本音響学会講演論文集, 3-2-6, pp. 587–588, 6 1979.

[32] 三輪譲二, 小原政彦, 牧野正三, 城戸健一. 非線形スペクトルマッチングによる単語音声認識の一方式. 電子情報通信学会論文誌, Vol. J64-D, No. 1, pp. 46–53, 1981.

[33] Esther Levin and Roberto Pieraccini. Dynamic planar warping for optical character recognition. In *Proceedings of IEEE International Conference on Acoustics, Speech & Signal Processing*, pp. III 149–152, 1992.

[34] Oscar E. Agazzi and Shyh-shiaw Kuo. Hidden Markov model based optical character recognition in the presence of deterministic transformations. *Pattern Recognition*, Vol. 26, No. 12, pp. 1813–1826, 1993.

[35] Oscar E.Agazzi, Shyh-shiaw Kuo, Esther Levin, and Roberto Pieraccini. Connected and degraded text recognition using planar hidden Markov models. In *Proceedings of IEEE International Conference on Acoustics, Speech & Signal Processing*, pp. V113–116, 1993.

[36] Shyh-shiaw Kuo and Oscar E.Agazzi. Keyword spotting in poorly printed documents using pseudo 2-d hidden Markov models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 16, No. 8, pp. 842–848, 1994.

[37] 杉村昌彦，飯国洋二，足立紀彦. ツェルニケモーメントを特徴量とする 2 次元動的計画法を用いたイメージマッチング. 電子情報通信学会論文誌, Vol. J80-D-II, No. 1, pp. 101–108, 1997.

[38] 磯道義典, 小川辰男. 動的計画法によるパターン ・マッチング. 情報処理, Vol. 16, No. 1, pp. 15–22, 1975.

[39] 迫江博昭, 千葉成美. 動的計画法を利用した音声の時間正規化に基づく連続音声認識. 日本音響学会誌, Vol. 27, No. 9, pp. 483–490, 9 1971.

[40] Hiroaki Sakoe and Seibi Chiba. Dynamic programming algorithm optimization for spoken word recognition. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, Vol. 26, No. 1, pp. 43–49, 2 1978.

[41] Roger K.Moore. A dynamic programming algorithm for the distance between two finite areas. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. PAMI-1, No. 1, pp. 86–88, 1979.

[42] 田中栄一, 菊池良昭. 図形間の距離. 電子情報通信学会論文誌, Vol. J63-D, No. 12, pp. 1018–1025, 1980.

[43] Eiichi Tanaka. A two dimensional context-dependent similarity measure. *The Transactions of the IEICE of Japan*, Vol. E.68, No. 10, pp. 667–673, 1985.

[44] C.M.Wu, R.M.Owens, and M.J.Irwin. Distortion processing in image matching problems. In *Proceedings of IEEE International Conference on Acoustics, Speech & Signal Processing*, pp. 2181–2184, 1990.

[45] 山田博三. 動的計画法による画像間の弾性マッチング法 - ピクセル DP 整合法. 電子情報通信学会技術研究報告, 1993.

[46] 大田友一, 山田博三. 動的計画法によるパターンマッチング. 情報処理, Vol. 30, No. 9, pp. 1058–1066, 1989.

[47] 愼　重弼, 迫江博昭. 筆順 ・画数自由オンライン文字認識のための画対応決定法 – 多層キューブサーチ –. 電子情報通信学会論文誌, (掲載予定).

[48] Thomas L. Morin and Roy E. Marsten. Branch-and-bound strategies for dynamic programming. *Operations Research*, Vol. 24, No. 4, 7-8 1976.

[49] Martin A. Fischler and Robert A. Elschlager. The representation and matching of pictorial structures. *IEEE Transactions on Computers*, Vol. C-22, No. 1, 1973.

[50] Haluk Derin and Howard Elliott. Modeling and segmentation of noisy and textured images using Gibbs random fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 9, No. 1, pp. 39–55, 1987.

[51] Alberto Martelli and Ugo Montanari. Optimal smoothing in picture processing: an application to fingerprints. In *Information Processing 71 Proceedings of the IFIP Congress 1971*, Vol. 1. North-Holland Publishing Co., Amsterdam, Netherlands, 1972.

[52] Ugo Montanari. On the optimal approximation of discrete functions with low-dimensional tables. In *Information Processing 71 Proceedings of the IFIP Congress 1971*, Vol. 2. North-Holland Publishing Co., Amsterdam, Netherlands, 1972.

[53] 迫江博昭, 藤井浩美, 吉田和永, 亘理誠夫. フレーム同期化, ビームサーチ, ベクトル量子化の統合による DP マッチングの高速化. 電子情報通信学会論文誌, Vol. J71-D, No. 9, pp. 1650–1659, 9 1988.

[54] H. Ney, D. Mergel, A. Noll, and A. Paeseler. Data driven search organization for continuous speech recognition. *IEEE Transactions on Signal Processing*, Vol. 40, No. 2, pp. 272–281, 2 1992.

[55] 迫江博昭, 片山喜規, 内田誠一. 文音声認識を目的とした CYK 解析アルゴリズムの効率化. 九州大学大学院システム情報科学研究科報告, Vol. 1, No. 1, pp. 95–100, 1996.

[56] 高橋伸弥, 迫江博昭. 雑音画像中の多元信号軌跡の解析. 電子情報通信学会論文誌, Vol. J80-D-II, No. 11, pp. 3011–3019, 11 1998.

[57] 山本芳嗣, 久保幹雄. 巡回セールスマン問題への招待, シリーズ 〈現代人の数理〉, 第 12 巻. 朝倉書店, 1997.

[58] 奥村晴彦. C 言語による最新アルゴリズム事典, ソフトウェアテクノロジー, 第 13 巻. 技術評論社, 1991.

[59] 浅田　稔. ダイナミックシーンの理解. 電子情報通信学会, 1994.

[60] Berthold K.P. Horn and Brian G. Schunck. Determining optical flow. *Artificial Intelligence*, Vol. 17, No. 1-3, pp. 185–203, 8 1981.

[61] 森　俊二, 山本和彦, 山田博三, 斎藤泰一. 手書教育漢字のデータベースについて. 電子技術総合研究所彙報, Vol. 43, No. 11,12, pp. 60–81, 1979.

[62] 水上嘉樹, 古賀和利. 線素方向特徴を用いた変位抽出を行う手書き漢字認識システム. 電子情報通信学会技術研究報告, PRU96-188, 1997.

付録 A

DP 漸化式の導出

本章では，第 4，5章での議論の基本となっている DP 漸化式が単調連続 2 次元ワープ決定問題を正しく解くものであることを示すために，実際に (2.2) 式からそれら DP 漸化式を導出する．これによりカラムワイズ DP アルゴリズムおよびピクセルワイズ DP アルゴリズムの正当性が証明される．

準備として次の定義をする．

$$d'(i,j,z_{i,j},z_{i-1,j},z_{i,j-1}) = \begin{cases} |a(i,j)-b(x(i,j),y(i,j))| & \text{if } z_{i,j},z_{i-1,j},z_{i,j-1} \text{ satisfy (2.3)}\sim\text{(2.6)} \\ \infty & \text{otherwise} \end{cases} \tag{A.1}$$

ここで $z_{i,j}=(x(i,j),y(i,j))$ とする．この定義により，(2.2) 式の制約条件つき最適化問題を次の制約条件無し問題に書き換えることができる．

$$D(\boldsymbol{A},\boldsymbol{B}) = \min_{z_{1,1}\overset{\text{rs}}{\rightarrow}z_{N,N}} \sum_{i=1}^{N}\sum_{j=1}^{N} d'(i,j,z_{i,j},z_{i-1,j},z_{i,j-1}) \tag{A.2}$$

ここで記号 $\overset{\text{rs}}{\rightarrow}$ はラスタスキャン順の記号列を意味する．例えば，

$$z_{i-1,j+1} \overset{\text{rs}}{\rightarrow} z_{i,j} \;\Leftrightarrow\; z_{i-1,j+1}, z_{i-1,j+2}, \ldots, z_{i-1,N}, z_{i,1}, z_{i,2}, \ldots, z_{i,j}$$

とする．以下，**A.1**ではカラムワイズ DP アルゴリズムが基づく DP 漸化式，**A.2**ではピクセルワイズ DP アルゴリズムが基づく DP 漸化式をそれぞれ導出する．

A.1 カラムワイズ DP アルゴリズムが基づく DP 漸化式の導出

(A.2) 式は以下のように書き直せる．

$$D(\boldsymbol{A},\boldsymbol{B}) = \min_{z_{1,1} \overset{\mathrm{rs}}{\to} z_{N,N}} \left[g(z_{1,1} \overset{\mathrm{rs}}{\to} z_{1,N}) + \sum_{(k,l)=(2,1)\overset{\mathrm{rs}}{\to}(N,N)} d'(k,l,z_{k,l},z_{k-1,l},z_{k,l-1}) \right] \tag{A.3}$$

ここで

$$g(z_{1,1} \overset{\mathrm{rs}}{\to} z_{1,N}) = \sum_{j=1}^{N} d'(1,j,z_{1,j},*,z_{1,j-1})$$

さらに，右辺第 2 項の総和を変数 $z_{1,1} \overset{\mathrm{rs}}{\to} z_{1,N}$ に関係する部分とそれ以外に分解すると，上式は以下のように書き直せる．

$$\begin{aligned}
D(\boldsymbol{A},\boldsymbol{B}) &= \min_{z_{1,1} \overset{\mathrm{rs}}{\to} z_{N,N}} \Bigg[g(z_{1,1} \overset{\mathrm{rs}}{\to} z_{1,N}) + \sum_{(k,l)=(2,1)\overset{\mathrm{rs}}{\to}(2,N)} d'(k,l,z_{k,l},z_{k-1,l},z_{k,l-1}) \\
&\qquad + \sum_{(k,l)=(3,1)\overset{\mathrm{rs}}{\to}(N,N)} d'(k,l,z_{k,l},z_{k-1,l},z_{k,l-1}) \Bigg] \\
&= \min_{z_{2,1} \overset{\mathrm{rs}}{\to} z_{N,N}} \Bigg[\min_{z_{1,1} \overset{\mathrm{rs}}{\to} z_{1,N}} \Big[g(z_{1,1} \overset{\mathrm{rs}}{\to} z_{1,N}) + \sum_{(k,l)=(2,1)\overset{\mathrm{rs}}{\to}(2,N)} d'(k,l,z_{k,l},z_{k-1,l},z_{k,l-1}) \Big] \\
&\qquad + \sum_{(k,l)=(3,1)\overset{\mathrm{rs}}{\to}(N,N)} d'(k,l,z_{k,l},z_{k-1,l},z_{k,l-1}) \Bigg] \\
&= \min_{z_{2,1} \overset{\mathrm{rs}}{\to} z_{N,N}} \Bigg[g(z_{2,1} \overset{\mathrm{rs}}{\to} z_{2,N}) \\
&\qquad + \sum_{(k,l)=(3,1)\overset{\mathrm{rs}}{\to}(N,N)} d'(k,l,z_{k,l},z_{k-1,l},z_{k,l-1}) \Bigg]
\end{aligned} \tag{A.4}$$

以上の操作は変数 $z_{1,1} \overset{\mathrm{rs}}{\to} z_{1,N}$ の削除 (elimination) と呼ばれる．この操作を繰り返していけば，次の一般式が得られる．

$$D(\boldsymbol{A},\boldsymbol{B}) = \min_{z_{i,1} \overset{\mathrm{rs}}{\to} z_{N,N}} \Bigg[g(z_{i,1} \overset{\mathrm{rs}}{\to} z_{i,N}) + \sum_{(k,l)=(i+1,1)\overset{\mathrm{rs}}{\to}(N,N)} d'(k,l,z_{k,l},z_{k-1,l},z_{k,l-1}) \Bigg] \tag{A.5}$$

ここで，

$$g(z_{i,1} \overset{\mathrm{rs}}{\to} z_{i,N}) = \min_{z_{i-1,1} \overset{\mathrm{rs}}{\to} z_{i-1,N}} \Bigg[g(z_{i-1,1} \overset{\mathrm{rs}}{\to} z_{i-1,N}) + \sum_{(k,l)=(i,1) \overset{\mathrm{rs}}{\to} (i,N)} d'(k,l,z_{k,l},z_{k-1,l},z_{k,l-1}) \Bigg] \tag{A.6}$$

この (A.6) 式を $i=1$ から順 $i=N$ まですべての $z_{i,1} \overset{\mathrm{rs}}{\to} z_{i,N}$ の組合せについて計算していけば，最終的に

$$D(\boldsymbol{A},\boldsymbol{B}) = \min_{z_{N,1} \overset{\mathrm{rs}}{\to} z_{N,N}} g(z_{N,1} \overset{\mathrm{rs}}{\to} z_{N,N}) \tag{A.7}$$

として最適解が求まることは明らかである．

ところで，(A.6) 式右辺の $g(z_{i,1} \overset{\mathrm{rs}}{\to} z_{i,N})$ についてはすべての $z_{i,1} \overset{\mathrm{rs}}{\to} z_{i,N}$ の組合せのうちで単調連続性を満たすものだけについて計算すればよいから，$z_{i,1} \overset{\mathrm{rs}}{\to} z_{i,N}$ は $\boldsymbol{xy}(i)$ と置き換えてよい．また (A.6) 式左辺の最小値選択において，$(z_{i-1,1} \overset{\mathrm{rs}}{\to} z_{i-1,N})$ と $\boldsymbol{xy}(i)$ との間で単調連続性が満たされない場合は括弧内が無限大となるので考える必要はない．以上から (A.6) 式は次のように整理される．

$$\begin{aligned} g(\boldsymbol{xy}(i)) &= \min_{\overline{\boldsymbol{xy}} \in \overline{XY}(\boldsymbol{xy}(i))} \left[g(\overline{\boldsymbol{xy}}) + \sum_{(k,l)=(i,1) \overset{\mathrm{rs}}{\to} (i,N)} |a(k,l) - b(x(k,l),y(k,l))| \right] \\ &= d(i,\boldsymbol{xy}(i)) + \min_{\overline{\boldsymbol{xy}} \in \overline{XY}(\boldsymbol{xy}(i))} g(\overline{\boldsymbol{xy}}) \end{aligned} \tag{A.8}$$

上式はまさにカラムワイズ DP アルゴリズムの漸化式である．よってカラムワイズ DP アルゴリズムが正しく (2.2) 式の単調連続 2 次元ワープ決定問題を解くことが証明された．なお，**3.2**での議論を参考にすれば，この漸化式から **4.2**で説明した多段決定過程 (図 4.1) が構成されるのは明らかである．

A.2 ピクセルワイズ DP アルゴリズムが基づく DP 漸化式の導出

(A.4) 式右辺第 2 項の総和のうち $z_{1,1}$に関するのは $d'(2,1,z_{2,1},z_{1,1},*)$ だけであることを考慮すれば，(A.4) 式は次のように書き直せる．

$$\begin{aligned}
D(\boldsymbol{A},\boldsymbol{B}) &= \min_{z_{1,1}\overset{\mathrm{rs}}{\to}z_{N,N}} \Bigg[g(z_{1,1}\overset{\mathrm{rs}}{\to} z_{1,N}) + d'(2,1,z_{2,1},z_{1,1},*) \\
&\qquad + \sum_{(k,l)=(2,2)\overset{\mathrm{rs}}{\to}(N,N)} d'(k,l,z_{k,l},z_{k-1,l},z_{k,l-1}) \Bigg] \\
&= \min_{z_{1,2}\overset{\mathrm{rs}}{\to}z_{N,N}} \Bigg[\min_{z_{1,1}} \Big[g(z_{1,1}\overset{\mathrm{rs}}{\to} z_{1,N}) + d'(2,1,z_{2,1},z_{1,1},*) \Big] \\
&\qquad + \sum_{(k,l)=(2,2)\overset{\mathrm{rs}}{\to}(N,N)} d'(k,l,z_{k,l},z_{k-1,l},z_{k,l-1}) \Bigg] \\
&= \min_{z_{1,2}\overset{\mathrm{rs}}{\to}z_{N,N}} \Bigg[g(z_{1,2}\overset{\mathrm{rs}}{\to} z_{2,1}) \\
&\qquad + \sum_{(k,l)=(2,2)\overset{\mathrm{rs}}{\to}(N,N)} d'(k,l,z_{k,l},z_{k-1,l},z_{k,l-1}) \Bigg] \qquad \text{(A.9)}
\end{aligned}$$

以上の操作は変数 $z_{1,1}$の削除に相当する．この操作を繰り返していけば，次の一般式が得られる．

$$\begin{aligned}
D(\boldsymbol{A},\boldsymbol{B}) = \min_{z_{i-1,j+1}\overset{\mathrm{rs}}{\to}z_{N,N}} \Bigg[& g(z_{i-1,j+1}\overset{\mathrm{rs}}{\to} z_{i,j}) \\
& + \sum_{(k,l)=(i,j+1)\overset{\mathrm{rs}}{\to}(N,N)} d'(k,l,z_{k,l},z_{k-1,l},z_{k,l-1}) \Bigg] \qquad \text{(A.10)}
\end{aligned}$$

ここで

$$g(z_{i-1,j+1}\overset{\mathrm{rs}}{\to} z_{i,j}) = \min_{z_{i-1,j}} \Big[g(z_{i-1,j}\overset{\mathrm{rs}}{\to} z_{i,j-1}) + d'(i,j,z_{i,j},z_{i-1,j},z_{i,j-1}) \Big] \qquad \text{(A.11)}$$

前節の議論と同様に，$z_{i-1,j+1}\overset{\mathrm{rs}}{\to} z_{i,j}$ の組合せのうち単調連続性を満たす$\boldsymbol{xy}(i,j)$ だけを考え，最小値選択時に $z_{i-1,j}$ のうち $z_{i,j}$，$z_{i-1,j+1}$ と単調連続性を満たす $z'_{i-1,j}$ だけを考えれば，上式は

$$g(\boldsymbol{xy}(i,j)) = \min_{z'_{i-1,j}} \Big[g(z'_{i-1,j}, z_{i-1,j+1}\overset{\mathrm{rs}}{\to} z_{i,j-1}) + |a(i,j) - b(x(i,j),y(i,j))| \Big]$$

$$
\begin{aligned}
&= |a(i,j) - b(x(i,j), y(i,j))| + \min_{z'_{i-1,j}} g(z'_{i-1,j}, z_{i-1,j+1} \stackrel{\mathrm{rs}}{\rightarrow} z_{i,j-1}) \\
&= d(i,j,\boldsymbol{xy}(i,j)) + \min_{\overline{\boldsymbol{xy}} \in \overline{XY}(\boldsymbol{xy}(i,j))} g(\overline{\boldsymbol{xy}}) \qquad \text{(A.12)}
\end{aligned}
$$

と書き直される．これはまさにピクセルワイズ DP アルゴリズムの漸化式である．よってピクセルワイズ DP アルゴリズムが正しく (2.2) 式の単調連続 2 次元ワープ決定問題を解くことが証明された．この DP 漸化式から **5.2**で説明した多段決定過程 (図 5.1) が構成されるのは明らかである．